Rational Convex Programs, Their Feasibility, and the Arrow-Debreu Nash Bargaining Game

Vijay V. Vazirani[*]

Abstract

Over the last decade, combinatorial algorithms have been obtained for exactly solving several nonlinear convex programs. We first provide a formal context to this activity by introducing the notion of *rational convex programs* – this also enables us to identify a number of questions for further study. So far, such algorithms were obtained for total problems only. Our main contribution is developing the methodology for handling non-total problems, i.e., their associated convex programs may be infeasible for certain settings of the parameters.

The specific problem we study pertains to a Nash bargaining game, called **ADNB**, which is derived from the linear case of the Arrow-Debreu market model. We reduce this game to computing an equilibrium in a new market model called *flexible budget market*, and we obtain primal-dual algorithms for determining feasibility, as well as giving a proof of infeasibility and finding an equilibrium. We give an application of our combinatorial algorithm for **ADNB** to an important "fair" throughput allocation problem on a wireless channel.

[*] College of Computing, Georgia Institute of Technology, Atlanta, GA 30332–0280, E-mail: vazirani@cc.gatech.edu.

1 Introduction

The fascinating question of computability of market equilibria, which has been studied extensively within TCS over the last decade, has provided the area of algorithms a new direction, namely the design of efficient combinatorial algorithms[1] for exactly solving nonlinear convex programs. For this purpose, the classical primal-dual paradigm was suitably extended from its usual setting of LP-duality theory to convex programs and the KKT conditions; this task was initiated in [DPSV08].

We note that all problems attacked so far were total, i.e., their convex programs always have finite optimal solutions. The main contribution of this paper is to develop the methodology for handling problems that are not guaranteed to always have a solution, i.e., their convex programs may be infeasible for certain settings of the parameters. It turns out that in the setting of LP's, the primal-dual algorithms for non-total problems are not more involved than those for total problems. We illustrate this by comparing the algorithms for the problems of maximum weight perfect matching and maximum weight matching in bipartite graphs in Section 10. However, the situation is quite different for convex programs; see Section 1.2 for a high level description and Section 10 for a detailed analysis.

The specific question that led to these ideas was combinatorially solving Nash bargaining games. Nash bargaining [Nas50] is a central solution concept within game theory for "fair" allocation of utility among competing players in the presence of complete information; it has numerous applications and a large following, e.g., see [Kal85, TL89, OR94]. The solution to a Nash bargaining game is obtained by maximizing a concave function over a convex set, i.e., it is the solution to a convex program. If the conditions for efficiently running the ellipsoid algorithm hold [GLS88], in particular, if efficient separation oracles can be implemented for its constraints and objective function, then the solution can be obtained to any required degree of accuracy in polynomial time. Instead, we resorted to the design of combinatorial algorithms because they have several advantages over "continuous" algorithms, see Section 1.1.

The specific Nash bargaining game we study in this paper is called the *Arrow-Debreu Nash Bargaining Game*, abbreviated **ADNB**. The setup is the same as the linear case of the Arrow-Debreu market model, but instead of resorting to the solution concept of a market equilibrium for reallocating goods among the agents, we resort to a Nash bargaining solution. Interestingly enough, our algorithm for solving **ADNB** reduces it to a new, natural market model, which we call the *flexible budget market*.

In the next section, we first provide a formal, mathematical context to the new algorithmic activity mentioned above. This also enables us to identify a number of questions for further study, see Section 12.

[1] Informally, an algorithm that conducts a search over a discrete space. A key distinction is that whereas a continuous method gives a way of solving the LP or convex program underlying a given instance, and provides no insight about the problem itself, a combinatorial algorithm handles all instances of the specific problem by exploiting its special combinatorial structure, e.g., see [Vaz10c] for a detailed argument.

1.1 Rational convex programs and combinatorial algorithms for them

A combinatorial algorithm for exactly solving a convex program is possible only if it admits rational solutions. Starting with the classic Eisenberg-Gale convex program [EG59], whose solution yields an equilibrium for the linear case of Fisher's market model [DPSV08], in recent years, researchers have identified several interesting convex programs that always admit rational solutions. First, let us formally define this class. We will say that a nonlinear convex program is *rational* if, for any setting of its parameters to rational numbers such that it has a finite optimal solution, it admits an optimal solution that is rational and can be written using polynomially many bits in the number of bits needed to write all the parameters.

This class turns out to be surprisingly rich – besides several Nash bargaining games [Vaz10a], it captures the Fisher market model under several classes of utility functions: linear [EG59, DPSV08], utility functions defined via combinatorial problems, including some in Kelly's [Kel97] resource allocation model [JV08, CDV], spending constraint utilities [Vaz10b, BDX10], and piecewise-linear concave utilities in a market model that allows for perfect price discrimination [GV10]. It also captures the linear case of the Arrow-Debreu market model (this follows from the convex program of [Jai07] and a proof of rationality in [Eav76]).

Of course, rationality does not guarantee the existence of a combinatorial polynomial time algorithm. However, it turns out that such algorithms are known for solving almost all of the rational convex programs mentioned above. The only exceptions are 2-agent markets and Nash bargaining games given in [CDV] and [Vaz10a], and the linear case of Arrow-Debreu markets. For the former, polynomial time algorithms are given, in [CDV] and [Vaz10a], using only an LP-solver. This leads to a tantalizing question: is the class of rational convex programs solvable in this manner, i.e., by polynomial time algorithms that use only an LP-solver?

Both algorithmic approaches, continuous and combinatorial, are valuable in their own right and have different advantages. Indeed, it is the synergy between these two approaches that makes algorithm design a potent field. We point out the advantages of the latter approach below.

We believe that the case for designing combinatorial algorithms for rational convex programs is as compelling as that for integral LP's, i.e., linear programs that always admit integral optimal solutions. The latter led to deep combinatorial insights and constitute a substantial part of the field of combinatorial optimization. In recent years, insights gained from combinatorial algorithms for convex programs have also led to major progress. For instance, the recent proof of membership in PPAD of markets under piecewise-linear concave utilities [VY10] followed from a new, combinatorial way of characterizing equilibria [DPSV08] and helped settle, together with [CDDT09, CT09], the long-standing open problem of determining the exact complexity of this key case of markets.

Combinatorial algorithms also have several advantages over continuous algorithms in applications. For instance, recently Nisan et. al., faced with the problem of designing an auction system for Google for TV ads, converged to a market equilibrium based method, after exploring several different options [NBC$^+$09]. As stated by Nisan [Nis09], the actual implementation of this algorithm was inspired by combinatorial market equilibrium algorithms, which in turn solve convex programs combinatorially. Indeed, the easy adaptability of combinatorial algorithms to the special idiosyncrasies of an application often makes them the preferred method. In Section 4 we present an application of our combinatorial algorithm for **ADNB** to a throughput allocation problem on a wireless channel.

Although our algorithm is not strongly polynomial, recent results provide a reason to believe that our methodology could lead to it. Using scaling techniques, Orlin [Orl10] extended the DPSV algorithm to a strongly polynomial one, and an obvious open question is to extend the rest of the algorithms for rational convex programs as well.

1.2 Determining feasibility of a convex program

A linear program is infeasible iff no point satisfies all its constraints. In the case of a convex program, infeasibility could arise because of a second reason, namely the objective function is undefined at each point that satisfies all constraints. The convex program for **ADNB** exhibits only the second type of infeasibility; it always has points satisfying all constraints.

Even for the case of total problems, the primal-dual method operates in fundamentally different ways in the two settings of integral linear programs and rational convex programs; for details, see Section 4 in [DPSV08] or Section 10 of the present paper. This difference manifests itself even more acutely when we move to non-total problems, as shown in detail in Section 10.

For solving **ADNB**, we give a special procedure, Stage I in Algorithm 1, that tests for feasibility. If the instance is found to be infeasible, this procedure finds a proof of infeasibility. Otherwise, it yields special prices, called feasible prices, which provide Stage II with the right starting point for finding a solution to the given instance.

We give two different proofs of infeasibility. The first one uses an explicit dual of the convex program for **ADNB**, found by Devanur [Dev10]; for proving infeasibility, our procedure shows that this dual is unbounded. Often it is difficult to find an explicit dual of a convex program. For this reason, we give a second proof of infeasibility. We give an LP whose optimal solution is positive iff the given instance is feasible. Therefore, a dual feasible solution of value at most zero is a proof of infeasibility. Our procedure also finds such a proof of infeasibility.

1.3 New insights into balanced flows

Differences between complementary slackness conditions for an LP and the KKT conditions for a convex program (see Section 4 in [DPSV08]) leads to new difficulties in the latter setting. The new algorithmic idea of balanced flows, introduced in [DPSV08], helps overcome these difficulties. Although this notion yields the desired efficient algorithm, the use of the l_2 norm in the potential function used for proving polynomial time termination makes the proofs quite difficult. [DPSV08] observe that the l_2 norm can be dispensed with for defining balanced flows and ask, "Can a polynomial running time be established for the algorithm using the alternative definition, thereby dispensing with the l_2 norm altogether? At present we see no way of doing this ... ."

In Section 11 we answer this question in the negative by providing an infinite family of examples in which the natural l_1 norm-based potential function makes inverse exponentially small progress whereas the l_2 norm-based potential function makes inverse polynomial progress.

We observe that the combinatorial algorithms of [Vaz10b] and [GV10], which solve rational convex programs, also use the notion of balanced flows crucially. Balanced flows play an even more fundamental role in our work. As detailed in Section 8.4, they are used in several crucial ways in both stages of our algorithm. In addition, they are also used for defining the central notion of feasible prices.

2 The Nash Bargaining Game

An *n-person Nash bargaining game* consists of a pair $(\mathcal{N}, \boldsymbol{c})$, where $\mathcal{N} \subseteq \mathbf{R}_+^n$ is a compact, convex set and $\boldsymbol{c} \in \mathcal{N}$. Set $\mathcal{N}$ is the *feasible set* and its elements give utilities that the n players can simultaneously accrue. Point $\boldsymbol{c}$ is the *disagreement point* – it gives the utilities that the n players obtain if they decide not to cooperate. The set of n agents will be denoted by B and the agents will be numbered $1, 2, \ldots n$. Game $(\mathcal{N}, \boldsymbol{c})$ is said to be *feasible* if there is a point $\boldsymbol{v} \in \mathcal{N}$ such that $\forall i \in B, \; v_i > c_i$, and *infeasible* otherwise.

The solution to a feasible game is the point $\boldsymbol{v} \in \mathcal{N}$ that satisfies the following four axioms:

1. **Pareto optimality:** No point in $\mathcal{N}$ can weakly dominate $\boldsymbol{v}$.
2. **Invariance under affine transformations of utilities:** If the utilities of any player are redefined by multiplying by a scalar and adding a constant, then the solution to the transformed game is obtained by applying these operations to the particular coordinate of $\boldsymbol{v}$.
3. **Symmetry:** If the players are renumbered, then it suffices to renumber the coordinates of $\boldsymbol{v}$ accordingly.
4. **Independence of irrelevant alternatives:** If $\boldsymbol{v}$ is the solution for $(\mathcal{N}, \boldsymbol{c})$, and $\mathcal{S} \subseteq \mathbf{R}_+^n$ is a compact, convex set satisfying $\boldsymbol{c} \in \mathcal{S}$ and $\boldsymbol{v} \in \mathcal{S} \subseteq \mathcal{N}$, then $\boldsymbol{v}$ is also the solution for $(\mathcal{S}, \boldsymbol{c})$.

Via an elegant proof, Nash proved:

Theorem 1 Nash *[Nas50] If game $(\mathcal{N}, \boldsymbol{c})$ is feasible then there is a unique point in $\mathcal{N}$ satisfying the axioms stated above. This is also the unique point that maximizes $\Pi_{i \in B}(v_i - c_i)$, over all $\boldsymbol{v} \in \mathcal{N}$.*

Most papers in game theory assume that the given Nash bargaining game $(\mathcal{N}, \boldsymbol{c})$ is feasible. However, in this paper, it will be more natural to not make this assumption and to determine this fact algorithmically. Thus, we can have one of 2 outcomes:

Thus Nash's solution to his bargaining game involves maximizing a concave function over a convex domain, and is therefore the optimal solution to the following convex program.

$$\begin{aligned} \text{maximize} \quad & \sum_{i \in B} \log(v_i - c_i) && (1) \\ \text{subject to} \quad & \boldsymbol{v} \in \mathcal{N} \end{aligned}$$

As a consequence, if for a specific game, a separation oracle can be implemented in polynomial time, then using the ellipsoid algorithm one can get as good an approximation to the solution of this convex program as desired in time polynomial in the number of bits of accuracy needed [GLS88].

3 The Game ADNB

The game **ADNB**, short for *Arrow-Debreu Nash Bargaining game*, which will be studied extensively, is derived from the linear case of the Arrow-Debreu model. We state the latter first. Let $B = \{1, 2, \ldots, n\}$ be a set of agents and $G = \{1, 2, \ldots, g\}$ be a set of divisible goods. We will assume w.l.o.g. that there is a unit amount of each good. Let u_{ij} be the utility derived by agent i on receiving one unit of good j; w.l.o.g., we will assume that u_{ij} is integral. If x_{ij} is the amount of good j that agent i gets, for $1 \leq j \leq g$, then she derives total utility

$$v_i(x) = \sum_{j \in G} u_{ij} x_{ij}.$$

Finally, we assume that each agent has an initial endowment of these goods; for each good, the total amount possessed by the agents is 1 unit.

W.l.o.g. we may assume that each good is desired by at least one agent and each agent desires at least one good, i.e.,

$$\forall j \in G,\ \exists i \in B: \ u_{ij} > 0 \ \text{ and } \ \forall i \in B,\ \exists j \in G: \ u_{ij} > 0.$$

If not, we can remove the good or the agent from consideration.

The question is to find prices for these goods so that if each agent sells her entire initial endowment at these prices and uses the money to buy an optimal bundle of goods, the market clears exactly, i.e., there is no deficiency or surplus of any good. Such prices are called *equilibrium prices.*

The Arrow-Debreu market model gives one mechanism by which the agents can redistribute goods to achieve higher utilities. Another mechanism is to view this setup as a Nash bargaining game as follows. For each $i \in B$, let c_i denote the utility derived by agent i from her initial endowment. Regard this as agent i's disagreement utility and redistribute the goods in accordance with the Nash bargaining solution.

We will define game **ADNB** in a slightly more general manner: we will assume that the disagreement utilities, c_i's, are arbitrary numbers specified in the particular instance. (As stated in the Introduction, we will not deal with the nonsymmetric extension of **ADNB** in this paper.) Clearly, the Nash bargaining solution is the optimal solution to the following convex program:

$$\begin{aligned} &\text{maximize} && \sum_{i \in B} \log(v_i - c_i) && (2) \\ &\text{subject to} && \forall i \in B: \ v_i = \sum_{j \in G} u_{ij} x_{ij} \end{aligned}$$

$$\forall j \in G: \ \sum_{i \in B} x_{ij} \leq 1$$
$$\forall i \in B, \ \forall j \in G: \ x_{ij} \geq 0$$

The KKT conditions for this program are:

(1) $\forall j \in G: \ p_j \geq 0$.

(2) $\forall j \in G: \ p_j > 0 \Rightarrow \ \sum_{i \in B} x_{ij} = 1$.

(3) $\forall i \in B, \ \forall j \in G: p_j \geq \frac{u_{ij}}{v_i - c_i}$.

(4) $\forall i \in B, \ \forall j \in G: x_{ij} > 0 \Rightarrow p_j = \frac{u_{ij}}{v_i - c_i}$.

Theorem 2 *Program (2) is a rational convex program. Moreover, if it is feasible, then the dual solution is unique.*

Proof : We will show that if for a setting of rational parameters, program (2) is feasible, then the x_{ij}'s and p_j's are solutions to an LP and are therefore rational numbers that can be written using polynomially many bits. First, guess the x_{ij}'s that are non-zero in the optimal solution to program (2). Because of the assumption made on the instance, each p_j will be positive.

The variables of the LP will be the non-zero x_{ij}'s and for each good j, a new variable q_j, which is supposed to represent $1/p_j$. The LP will have the following constraints: for each q_j, there is one equation corresponding to the KKT condition (2), and for each nonzero x_{ij} there is one equation corresponding to the KKT condition (4). In addition, the LP has inequality constraints corresponding to KKT condition (3), for each $i \in B$ and $j \in G$. In all these constraints, v_i is replaced by $\sum_{j \in G} u_{ij} x_{ij}$. Finally, it has non-negativity constraints for all x_{ij}'s and q_j's. It is easy to check that all constraints are linear.

Since program (2) is feasible, so is the LP corresponding to the correct guess. The solution to this LP will satisfy all KKT conditions and hence is an optimal solution to program (2). This established rationality of the convex program. Finally, the strict concavity of the objective function of program (2) and the fact that the convex combination of any two of its feasible solutions is also feasible implies that the optimal values of v_i's is unique. Now, using the KKT condition (4), we get the uniqueness of p_j's as well. □

4 An Application to Throughput Allocation on a Wireless Channel

A central throughput allocation problem arising in the context of a wireless channel, such as in 3G technologies, is the following. There are n users $1, 2, \ldots n$ and the wireless router can be in any of m different states $1, 2, \ldots m$ whose probabilities, $\pi(j)$, can be estimated by sampling. Each user i derives utility at rate u_{ij} if it is connected to the router while the router is in state

j; the u_{ij}'s are known. No matter what state the router is in, only one user can be connected to it. If user i is given connection for $x_{ij} \leq \pi(j)$ of the time the router is in state j, for $1 \leq j \leq m$, then the total utility derived by i is $v_i = \sum_{j=1}^{m} u_{ij}x_{ij}$. Clearly, we must ensure the constraint $\sum_{i=1}^{n} x_{ij} \leq \pi(j)$, for each j. The question is to find a "fair" way of dividing the $\pi(j)$'s among the users.

The method of choice in the networking community is to use Kelly's proportional fair scheme [Kel97], which entails maximizing $\sum_i \log v_i$ subject to the constraints given above, i.e., solving the Eisenberg-Gale convex program [EG59]. Observe that the above setting can be viewed as a linear Fisher market with n users and m divisible goods. An elegant gradient descent algorithm for solving this convex program, given by David Tse [Tse] (see also [JPP00]), was implemented by Qualcomm in their chip sets and is used by numerous 3G wireless basestations [And09]. However, this solution may at times allocate unacceptably low utility to certain users. This was countered by giving users the ability to put a lower bound on channel rates, say c_i for user i. This enhanced problem was solved by changing the objective function of the convex program to maximizing $\sum_i \log(v_i - c_i)$; observe that this is precisely an instance of **ADNB**! However, now the gradient descent implementation ran into problems of instability, since it involved computing $u_{ij}/(v_i' - c_i)$, where v_i' is the current estimate of v_i; at intermediate points, the denominator may be too small or even negative.

A different solution, proposed and implemented by researchers at Lucent [AQS05], was to introduce the constraints $v_i > c_i$ in the Eisenberg-Gale program itself. The fairness guarantee achieved by this solution is unclear. Additionally, determining feasibility of the convex program now became a major issue [And09]. Instead, we have proposed experimenting with a heuristic adaptation of our combinatorial algorithm for **ADNB**, which will not have stability issues. As reported in the FOCS 2002 version of [DPSV08], an analogous heuristic adaptation of the DPSV algorithm was found to perform well on fairly large sized linear Fisher instances.

5 Fisher's Model and its Extension via Flexible Budgets

First we specify Fisher's market model for the case of linear utilities [BS00]. Consider a market consisting of a set of n buyers $B = \{1, 2, \ldots, n\}$, and a set of g divisible goods, $G = \{1, 2, \ldots, g\}$; we may assume w.l.o.g. that there is a unit amount of each good. Let m_i be the money possessed by buyer i, $i \in B$. Let u_{ij} be the utility derived by buyer i on receiving one unit of good j. Thus, if x_{ij} is the amount of good j that buyer i gets, for $1 \leq j \leq g$, then the total utility derived by i is

$$v_i(x) = \sum_{j=1}^{g} u_{ij}x_{ij}.$$

The problem is to find prices $\boldsymbol{p} = \{p_1, p_2, \ldots, p_g\}$ for the goods so that when each buyer is given her utility maximizing bundle of goods, the market clears, i.e., each good having a positive price is exactly sold, without there being any deficiency or surplus. Such prices are called *market clearing prices* or *equilibrium prices.*

The following is the Eisenberg-Gale convex program. Using the KKT conditions, one can show that its optimal solution is an equilibrium allocation for Fisher's linear market and the Lagrange variables corresponding to the inequalities give equilibrium prices for the goods (e.g., see Theorem 5.1 in [Vaz07]).

$$\begin{aligned}
\text{maximize} \quad & \sum_{i\in B} m_i \log v_i \qquad\qquad (3)\\
\text{subject to} \quad & \forall i \in B:\ v_i = \sum_{j\in G} u_{ij}x_{ij}\\
& \forall j \in G:\ \sum_{i\in B} x_{ij} \le 1\\
& \forall i \in B,\ \forall j \in G:\ x_{ij} \ge 0
\end{aligned}$$

Next, we introduce a flexible budget market as a modification of Fisher's linear case; this market will be used for solving **ADNB**. The utility functions of buyers are as before. The two main differences are that each buyer i now has a parameter c_i giving a strict lower bound on the amount of utility she wants to derive and buyers do not come to the market with a fixed amount of money, but instead the money they spend is a function of prices of goods in the following manner. Given prices $\boldsymbol{p}$ for the goods, define the *maximum bang-per-buck* of buyer i to be

$$\gamma_i = max_j \left\{ \frac{u_{ij}}{p_j} \right\}.$$

Now, buyer i's money is defined to be $m_i = 1 + \frac{c_i}{\gamma_i}$.

Let us say that set $S_i = \text{argmax}_j \left\{ \frac{u_{ij}}{p_j} \right\}$ constitutes i's *maximum bang-per-buck goods*. Clearly, at prices $\boldsymbol{p}$, any utility maximizing bundle of goods for i will consist of goods from S_i costing m_i money. Again the problem is to find market clearing or equilibrium prices. Observe that in an equilibrium, if it exists, each buyer i will derive utility exceeding c_i.

5.1 The Reduction

An instance I of **ADNB** is transformed to a flexible budget market $\mathcal{M}$ as stated above.

Theorem 3 *Instance I is feasible iff $\mathcal{M}$ is feasible. Moreover, if I and $\mathcal{M}$ are both feasible, then allocations $\boldsymbol{x}$ and dual $\boldsymbol{p}$ are optimal for I iff they are equilibrium allocations and prices for the flexible budget market $\mathcal{M}$.*

Proof : (⇒) First assume that I is feasible and that allocations $\boldsymbol{x}$ and dual $\boldsymbol{p}$ are optimal for RNB game I. Then I must satisfy the KKT conditions for convex program (2).

By the second KKT condition, each good having a positive price is fully sold. Assume that $y_{ij} > 0$. Then, by the definition of γ_i and the fourth KKT condition,

$$\gamma_i = \frac{u_{ij}}{p_j} = v_i - c_i.$$

The money of buyer i at prices $\boldsymbol{p}$ in market $\mathcal{M}$ is defined to be $m_i = 1 + c_i/\gamma_i$. The money spent by i in market $\mathcal{M}$ is:

$$\begin{aligned}\sum_{j\in G} x_{ij}p_j &= \sum_{j\in G} \frac{x_{ij}u_{ij}}{\gamma_i} \\ &= \frac{1}{v_i - c_i}\sum_{j\in G} x_{ij}u_{ij} = \frac{c_i}{v_i - c_i} = 1 + \frac{c_i}{v_i - c_i} = 1 + \frac{c_i}{\gamma_i} = m_i.\end{aligned}$$

Furthermore, by the third and fourth KKT conditions, i buys only her maximum bang-per-buck objects, thereby getting an optimal bundle. This proves that $\boldsymbol{x}$ and $\boldsymbol{p}$ constitute equilibrium allocations and prices for market $\mathcal{M}$.

($\Leftarrow$) Next, assume that $\mathcal{M}$ is feasible and that $\boldsymbol{x}$ and $\boldsymbol{p}$ are equilibrium allocations and prices for market $\mathcal{M}$. Now, $\boldsymbol{x}$ is clearly feasible for program (2); we will show that $\boldsymbol{x}$ and $\boldsymbol{p}$ satisfy all the KKT conditions for this program. The first two conditions are obvious.

Since i gets an optimal bundle of objects at prices $\boldsymbol{p}$,

$$x_{ij} > 0 \;\Rightarrow\; \frac{u_{ik}}{p_j} = \gamma_i.$$

Since i spends all her money,

$$m_i = 1 + \frac{c_i}{\gamma_i} = \sum_{j\in G} x_{ij}p_j = \sum_{k\in T_i} y_{ik}\frac{u_{ik}}{\gamma_i} = \frac{v_i}{\gamma_i}.$$

Therefore, $\gamma_i = v_i - c_i$. This gives the last two conditions as well. □

6 A Test for Equilibrium Prices

We will present an efficient algorithm for solving an instance I of the game **ADNB** by first reducing it to a flexible budget market, say $\mathcal{M}$. In this section, we first give an efficient algorithm for the following simpler question: Given prices $\boldsymbol{p} = \{p_1, \dots p_g\}$ for the goods in $\mathcal{M}$, determine if these are equilibrium prices, and if so, find an equilibrium allocation.

First, construct a directed network $N(\boldsymbol{p})$ as follows. $N(\boldsymbol{p})$ has a source s, a sink t, and vertex subsets B and G corresponding to the buyers and goods, respectively. For each good $j \in G$, there is an edge (s, j) of capacity p_j, and for each buyer $i \in B$, there is an edge (i, t) of capacity m_i, where $m_i = 1 + c_i/\gamma_i$ is i's money in $\mathcal{M}$. Recall that S_i contains i's maximum bang-per-buck goods. The edges between G and B are precisely the maximum bang-per-buck edges, i.e., those (j, i) such that $j \in S_i$. Each of these edges has infinite capacity.

Lemma 4 *Prices $\boldsymbol{p}$ are equilibrium prices for $\mathcal{M}$ iff the two cuts $(s, B \cup G \cup t)$ and $(s \cup B \cup G, t)$ are min-cuts in network $N(\boldsymbol{p})$. Moreover, if $\boldsymbol{p}$ are equilibrium prices, then the set of equilibrium allocations corresponds exactly to max-flows in $N(\boldsymbol{p})$.*

The proof of this lemma is straightforward using the transformation between a max-flow f in $N(\boldsymbol{p})$ and an allocation x in $\mathcal{M}$ given by $x_{ij} = f(j,i)/p_j$. The condition that $(s, B \cup G \cup t)$ and $(s \cup B \cup G, t)$ are min-cuts in network $N(\boldsymbol{p})$, and hence saturated by f, corresponds to all goods being sold and all buyers' money being spent. The fact that (j,i) is an edge in $N(\boldsymbol{p})$ iff $j \in S_i$ ensures that buyers get only their maximum bang-per-buck goods. Clearly, one max-flow computation suffices to determine if prices $\boldsymbol{p}$ are equilibrium prices for $\mathcal{M}$.

The next lemma gives the combinatorial object that yields equilibrium prices. Assume that $\boldsymbol{p}^*$ are equilibrium prices, i.e., $N(\boldsymbol{p}^*)$ satisfies the condition in Lemma 4. Let H be the uncapacitated directed subgraph of $N(\boldsymbol{p}^*)$ induced on $B \cup G$.

Lemma 5 *Given H, we can find $\boldsymbol{p}^*$ in strongly polynomial time.*

Proof : Consider the connected components of H after ignoring directions on its edges. In each component, pick a good and assign it price p, say. The prices of the rest of the goods in this component can be obtained in terms of p. The bang-per-buck, and hence the money, of each buyer in this component can also be obtained in terms of p. Finally, by equating the money of all buyers in this component with the total value of all goods in this component, we can compute p. □

Given two d-dimensional vectors with non-negative coordinates, $\boldsymbol{p}$ and $\boldsymbol{q}$, we will say that $\boldsymbol{p}$ *weakly dominates* $\boldsymbol{q}$ if for each coordinate j, $q_j \le p_j$. The following characterization will be useful in Algorithm 3.

Lemma 6 *Assume that market $\mathcal{M}$ is feasible and $\boldsymbol{p}$ is its unique equilibrium price vector. Let $\boldsymbol{q}$ be a vector of positive prices such that $(s, B \cup G \cup t)$ is a min-cut in network $N(\boldsymbol{q})$. Then $\boldsymbol{p}$ weakly dominates $\boldsymbol{q}$.*

Proof : Let us assume, for establishing a contradiction, that there are goods j such that $q_j > p_j$ and yet $(s, B \cup G \cup t)$ is a min-cut in $N(\boldsymbol{q})$. Let

$$\theta = \max_{j \in G} \left\{ \frac{q_j}{p_j} \right\} \qquad \text{and} \qquad S = \{ j \in G \mid q_j = \theta p_j \}.$$

Clearly, $\theta > 1$.

Let T_p and T_q be the set of buyers who are interested in goods in S at prices p and q, respectively. Since S represents the set of goods whose prices increase by the largest factor in going from prices $\boldsymbol{p}$ to $\boldsymbol{q}$, $T_q \subseteq T_p$. We claim that a buyer $i \in T_q$ is not interested in any goods in $G - S$ at prices $\boldsymbol{p}$ (because otherwise at prices $\boldsymbol{q}$, i will not be interested in any goods in S, since their prices increased the most). Therefore, in any max-flow in $N(\boldsymbol{p})$, all flow going through nodes in T_q must also have used nodes in S. Therefore,

$$\sum_{j \in S} p_j \geq \sum_{i \in T_q} \left(1 + \frac{c_i}{\gamma_i}\right),$$

where γ_i is the maximum bang-per-buck of buyer i w.r.t. prices $\boldsymbol{p}$. Multiplying this inequality by θ and using the fact that $\theta > 1$ we get,

$$\theta \sum_{j \in S} p_j \geq \sum_{i \in T_q} \left(\theta + \frac{c_i \theta}{\gamma_i}\right) > \sum_{i \in T_q} \left(1 + \frac{c_i \theta}{\gamma_i}\right),$$

Observe that the maximum bang-per-buck of buyer $i \in T_q$ w.r.t. prices $\boldsymbol{q}$ is γ_i/θ. Therefore, the last inequality implies that w.r.t. prices $\boldsymbol{q}$, the total value of goods in S is strictly more than the total value of money possessed by buyers in T_q. On the other hand, since in $N(\boldsymbol{q})$ all flow using nodes of T_q goes through S, we get that $(s, B \cup G \cup t)$ is not a min-cut in network $N(\boldsymbol{q})$, leading to a contradiction. □

Next assume that $\mathcal{M}$ is an arbitrary flexible budget market, not necessarily feasible. We will say that prices $\boldsymbol{q}$ are *small* if $\boldsymbol{q}$ is a positive vector and $(s, B \cup G \cup t)$ is a min-cut in network $N(\boldsymbol{q})$. Observe that in this case, each good j must have an edge (j, i), for some buyer i, incident at it. By Lemma 6, if $\mathcal{M}$ is feasible and prices $\boldsymbol{q}$ are small, then they are weakly dominated by the equilibrium prices, $\boldsymbol{p}$. Observe however that the contrapositive of Lemma 6 does not hold, i.e., $\boldsymbol{p}$ may dominate positive prices $\boldsymbol{q}$, yet $(s, B \cup G \cup t)$ may not be a min-cut in network $N(\boldsymbol{q})$.

The background given so far will suffice to read Section A, which gives an algorithm that converges to the solution of a given feasible instance of **ADNB** in the limit. This section may serve as a suitable warm-up, since our polynomial time algorithm is quite involved.

7 Determining Feasibility

In this section, we will address the question of determining whether the given flexible budget market, $\mathcal{M}$, is feasible. We will give a characterization of feasible markets and we will derive conditions that yield a proof of infeasibility.

Clearly, if we can find small prices $\boldsymbol{p}$ and a max-flow f in network $N(\boldsymbol{p})$ such that the flow gives each buyer i strictly more than c_i utility, then $\mathcal{M}$ is feasible. We first show, using the notion of balanced flows, that this test of feasibility is in fact a property of prices $\boldsymbol{p}$ only.

7.1 Balanced flows

We will follow the exposition in [Vaz07] and refer the reader to this chapter for all facts stated below without proof. For simplicity, denote the current network, $N(\boldsymbol{p})$, by simply N. Given a feasible flow f in N, let $R(f)$ denote the residual graph w.r.t. f. Define the *surplus* of buyer

i w.r.t. flow f in network N, $\theta_i(N,f)$, to be the residual capacity of the edge (i,t) w.r.t. flow f in network N, i.e., m_i minus the flow sent through the edge (i,t). The *surplus vector w.r.t. flow f* is defined to be $\theta(N,f) := (\theta_1(N,f), \theta_2(N,f), \ldots, \theta_n(N,f))$. Let $\|v\|$ denote the l_2 norm of vector v. A *balanced flow* in network N is a flow that minimizes $\|\theta(N,f)\|$. A balanced flow must be a max-flow in N because augmenting a given flow can only lead to a decrease in the l_2 norm of the surplus vector.

A balanced flow in N can be computed using at most n max-flow computations. It is easy to see that all balanced flows in N have the same surplus vector. Hence, for each buyer i, we can define $\theta_i(N)$ to be the surplus of i w.r.t. any balanced flow in N; we will shorten this to θ_i when the network is understood. The key property of a balanced flow that our algorithm will rely on is that a maximum flow f in N is balanced iff it satisfies Property 1:

Property 1: For any two buyers i and j, if $\theta_i(N,f) < \theta_j(N,f)$ then there is no path from node i to node j in $R(f) - \{s,t\}$.

Balanced flows play a crucial role in both stages of our algorithm; moreover, they have multiple uses. In Section 8.4, after stating the full algorithm, we state the various uses of this notion.

7.2 A characterization of feasibility

Let $\boldsymbol{p}$ be small prices and let $(\theta_1, \ldots, \theta_n)$ be the surplus vector of a balanced flow in $N(\boldsymbol{p})$. We will say that $\boldsymbol{p}$ are *feasible prices* if for each buyer i, $\theta_i < 1$.

Lemma 7 *Market $\mathcal{M}$ is feasible iff there are feasible prices for it.*

Proof : If $\mathcal{M}$ is feasible, its equilibrium prices are feasible, since for each buyer i, $\theta_i = 0$. Next, assume that $\boldsymbol{p}$ are feasible prices for $\mathcal{M}$. By definition, the flow sent on edge (i,t) in a balanced flow in $N(\boldsymbol{p})$ is

$$m_i - \theta_i > m_i - 1 = \left(1 + \frac{c_i}{\gamma_i}\right) - 1 = \frac{c_i}{\gamma_i}.$$

The utility accrued by i from this allocation is $\gamma_i(m_i - \theta_i) > c_i$. Hence $\mathcal{M}$ is feasible. □

Observe that if a max-flow f in network $N(\boldsymbol{p})$ gives each buyer i strictly more than c_i utility, then so will a balanced flow in $N(\boldsymbol{p})$. Hence, feasibility is a property of prices $\boldsymbol{p}$ only.

Rather than working with θ_i, it will sometimes be more convenient to work with $\theta_i - 1$. Hence, w.r.t. small prices $\boldsymbol{p}$, let us define the *1-surplus* of buyer i to be $\beta_i = \theta_i - 1$, where $(\theta_1, \ldots, \theta_n)$ is the surplus vector of a balanced flow in $N(\boldsymbol{p})$. Now, another definition of feasible prices is that they be small and for each buyer i, $\beta_i < 0$.

7.3 Proof of infeasibility

We will provide two ways of establishing infeasibility of the given market. The first is via the dual of an LP whose optimal solution tells us if $\mathcal{M}$ is feasible and the second is via the dual of convex program (2). Given an infeasible market, Stage I of our algorithm will terminate in a way that yields both proofs.

The game is feasible iff there is a point $v \in \mathcal{N}$ such that for each agent $i \in B$, $v_i > c_i$. In order to capture feasibility via a linear program, let us restate as follows: the game is feasible iff

$$\max_{v \in \mathcal{N}} \min_{i \in B} : \ (v_i - c_i) > 0.$$

Observe that the expression on the left hand side is the optimal objective function value of LP (4). Hence, the game is feasible iff the optimal solution to LP (4) is greater than zero. Clearly, this LP is maximizing t; however, in order to obtain a convenient dual, we will write it as minimizing $-t$:

$$\begin{aligned} \text{minimize} \quad & -t && (4) \\ \text{subject to} \quad & \forall i \in B: \ \sum_{j \in G} u_{ij} x_{ij} \geq c_i + t \\ & \forall j \in G: \ -\sum_{i \in B} x_{ij} \geq -1 \\ & \forall i \in B, \ \forall j \in G: \ x_{ij} \geq 0 \end{aligned}$$

Let y_i's and z_j's be the dual variables corresponding to the first and second set of inequalities, respectively. The dual program is:

$$\begin{aligned} \text{maximize} \quad & \sum_{i \in I} c_i y_i - \sum_{j \in G} z_j && (5) \\ \text{subject to} \quad & \forall i \in B, \ \forall j \in G: \ u_{ij} y_i - z_j \leq 0 \\ & \sum_{i \in B} y_i = 1 \\ & \forall i \in B: \ y_i \geq 0 \\ & \forall j \in G: \ z_j \geq 0 \end{aligned}$$

Lemma 8 *If there exist prices $\boldsymbol{p}$ s.t. $\sum_{j \in G} p_j > 0$ and $\sum_{i \in B} \beta_i \ \geq \ 0$, then the given game is infeasible.*

Proof : W.r.t. prices $\boldsymbol{p}$, compute the maximum bang-per-buck, γ_i, of each buyer i. By definition of maximum bang-per-buck,

$$\forall i \in B, \ \forall j \in G: \qquad \gamma_i \geq \frac{u_{ij}}{p_j}.$$

Let $\mu = \sum_{i \in B} 1/\gamma_i$. Since $\sum_{j \in G} p_j > 0, \ \mu > 0$.

Next, consider prices $\boldsymbol{q}$, where for each $j \in G$, $q_j = p_j/\mu$. Since all the prices have been scaled by the same factor, the network remains unchanged. Clearly, the maximum bang-per-buck of buyer i w.r.t. $\boldsymbol{q}$ is $\gamma'_i = \mu\gamma_i$ and

$$\forall i \in B,\ \forall j \in G : \qquad \gamma'_i \geq \frac{u_{ij}}{q_j}.$$

Let $y_i = 1/\gamma'_i$, for $i \in B$, and $z_j = q_j$, for $j \in G$. We will show that (y, z) is a feasible solution for the dual LP (5). The first set of inequalities is established by noting that

$$\forall i \in B,\ \forall j \in G : \qquad \gamma'_i \geq \frac{u_{ij}}{q_j} \quad \text{hence} \quad u_{ij}y_i \leq z_j.$$

Next we show that the equality constraint holds:

$$\sum_{i\in B} y_i = \sum_{i\in B} \frac{1}{\gamma'_i} = \left(\frac{1}{\mu}\right) \cdot \sum_{i\in B} \frac{1}{\gamma_i} = 1.$$

Let $\alpha'_i = c_i/\gamma'_i$ and let β'_i be the 1-surplus of buyer i w.r.t. prices $\boldsymbol{q}$. The objective function value of the dual solution (y, z) is

$$\sum_{i\in B} c_i y_i - \sum_{j\in G} z_j = \sum_{i\in B} \alpha'_i - \sum_{j\in G} q_j = \sum_{i\in B} \beta'_i \;=\; \left(\frac{1}{\mu}\right) \cdot \sum_{i\in B} \beta_i \geq 0.$$

Therefore, at optimality, $-t \geq 0$, i.e., $t \leq 0$, hence establishing infeasibility of the game. □

Next, we derive a condition under which the following convex program, which is the dual of (2), has an unbounded solution, hence proving infeasibility of the primal. This dual program was given by Devanur [Dev10]. Its variables are q_j's and y_i's.

$$\begin{aligned} \text{minimize} \quad & \sum_{j\in G} q_j - \sum_{i\in B} c_i y_i - \sum_{i\in B} \log(y_i) \qquad (6)\\ \text{subject to} \quad & \forall i \in B,\ \forall j \in G : \ q_j \geq u_{ij} y_i \end{aligned}$$

Lemma 9 *If there exist positive prices $\boldsymbol{p}$ such that network $N(\boldsymbol{p})$ can be partitioned into two, induced on (B', G') and $((B - B'), (G - G'))$, where $B' \subset B$ and $G' \subset G$, such that $\sum_{i\in(B-B')} \beta_i \ \geq\ 0$ and $\forall i \in (B - B'),\ \forall j \in G' :\ u_{ij} = 0$, then the given game is infeasible.*

Proof : Observe that

$$\sum_{i\in(B-B')} \beta_i \;=\; \left(\sum_{i\in(B-B')} \theta_i\right) - |B - B'| \;=\; \left(\sum_{i\in(B-B')} m_i\right) - \left(\sum_{j\in(G-G')} p_j\right) - |B - B'|$$

$$= \left(\sum_{i\in(B-B')} \frac{c_i}{\gamma_i}\right) - \left(\sum_{j\in(G-G')} p_j\right) \;\geq 0.$$

It is easy to see that setting q_j to p_j and y_i to $1/\gamma_i$ gives a feasible solution to program (6). Multiply the prices of all goods in $(G - G')$ by x and let $x \to \infty$. Observe that because of the condition $\forall i \in (B - B'),\ \forall j \in G' :\ u_{ij} = 0$, no new edges will be introduced in the network. Hence, the updated setting of q_j's and y_i's still yields a feasible solution to the dual.

As $x \to 0$, for each $i \in (B - B')$, $\log(\gamma_i) \to -\infty$. Also, $\sum_{j \in (G-G')} q_j - \sum_{i \in (B-B')} c_i y_i$ is either 0 or tends to $-\infty$. Hence the entire objective function tends to $-\infty$. Therefore, the dual is unbounded and hence the primal is infeasible. □

8 Details of the Algorithm for ADNB

We will impose the following condition throughout; by Lemma 6, it will ensure that prices are always small[2].

Invariant: W.r.t. current prices, $\boldsymbol{p}$, $(s, B \cup G \cup t)$ is a min-cut in network $N(\boldsymbol{p})$.

It is easy to see that the prices found by Initialization satisfy the Invariant.

Let f be a balanced flow in $N(\boldsymbol{p})$. Since the Invariant is always maintained, for each buyer i, $\theta_i \geq 0$ and hence $\beta_i \geq -1$. In the algorithm, we will change prices of a well-chosen set J of goods as follows. Multiply the price of each good in J by a variable x and initialize x to 1. In Stage I, we will decrease x and in Stage II we will raise x until the next event happens.

In the next lemma, we will assume that $J = G$ and we will study how the 1-surplus of buyers changes as a function of x. Define $x \cdot f$ to be the flow obtained by multiplying by x the flow on each edge w.r.t. f. Let $\beta_i(x)$ denote i's 1-surplus w.r.t. flow $x \cdot \boldsymbol{p}$. Let B' be the set of buyers having negative 1-surplus w.r.t. prices $\boldsymbol{p}$. If $B' = \emptyset$, define $b = \infty$ else define $b = \min_{i \in B'}\{-1/\beta_i\}$. Observe that in both cases, $b > 1$.

Lemma 10 *Flow $x \cdot f$ is a balanced flow in $N(x\boldsymbol{p})$ for $0 < x \leq b$, and for each $i \in B$, $\beta_i(x) = x\beta_i$.*

Proof : Since the Invariant holds and f is a max-flow in $N(\boldsymbol{p})$, the cut $(s, J \cup I \cup t)$ is saturated by f, and hence by $x \cdot f$ in $N(x\boldsymbol{p})$. Next we show that $x \cdot f$ is a feasible flow in $N(x\boldsymbol{p})$, i.e., for each buyer $i \in B$, edge (i, t) is not over saturated. Now, $\beta_i = \alpha_i - f(i, t)$. Therefore, the surplus on edge (i, t) w.r.t. flow $x \cdot f$ is $1 + x(\alpha_i - f(i, t)) = 1 + x\beta_i \geq 0$ for $0 < x \leq b$. Hence, edge (i, t) is not over saturated. Furthermore, $\beta_i(x) = x\beta_i$. Finally, since f satisfies Property 1 in $N(\boldsymbol{p})$, $x \cdot f$ satisfies it in $N(x\boldsymbol{p})$, thereby showing that it is a balanced flow. □

8.1 Details of Stage I

Algorithm 1 gives the pseudo code for Stage I. In this section, we give the subroutines used by this stage and Section 8.3 gives formal definitions of the predicates used in the While loops. A

[2] The following power point presentations may make the algorithm easier to understand:
http://www.cc.gatech.edu/ vazirani/Waterloo1.ppt
http://www.cc.gatech.edu/ vazirani/Waterloo2.ppt

run of Stage I is partitioned into *phases*, which are further partitioned into *iterations*. In Stage I, an iteration ends when a new edge is added to the network. A phase ends either when the condition of Step 7 holds or if for some $i \in I, \beta_i \geq 0$. In each iteration, the algorithm computes a balanced flow in the current network, $N(\boldsymbol{p})$.

We establish the following notation. For $J \subseteq G$, define $p(J) = \sum_{j \in J} p_j$ and $\Gamma(J) = \{i \in B \mid \exists j \in J \; s.t. \; (j,i) \in N(\boldsymbol{p})\}$. Similarly, for $I \subseteq B$, define $\alpha(I) = \sum_{i \in I} \alpha_i$, $m(I) = \sum_{i \in I} m_i$ and $\Gamma(I) = \{j \in G \mid \exists i \in B \; s.t. \; (j,i) \in N(\boldsymbol{p})\}$.

The sets B_c and G_c denote the *current sets of buyers and goods* being considered by the algorithm. These sets are initialized to B and G, respectively. At any point the algorithm, $B = B_c \cup B'$ and $G = G_c \cup G'$, where B' and G' are the sets of *adaptable buyers and goods*, respectively; their purpose is explained below. B' and G' are both initialized to $\emptyset$. As the algorithm proceeds, buyers are moved from B_c to B' and goods are moved from G_c to G'.

The subroutines used in Stage I are:

- **Find sets(I):** Sets $I \subseteq B_c$ and $J \subseteq G_c$ are initialized as follows.

  $$I \leftarrow \arg\min_{i \in B_c} \{\beta_i\} \quad \text{and} \quad J \leftarrow (\Gamma(I) - \Gamma(B_c - I)).$$

  Observe that J consists of goods that are the maximum bang-per-buck goods of buyers in I only. All edges from goods in $G_c - J$ to buyers in I are removed; this is justified in Lemma 15 below.

- **Update sets(I):** Find the set, I', of all buyers in $B_c - I$ such that there is a residual path from a buyer in I to a buyer in I'. Update

  $$I \leftarrow (I \cup I') \quad \text{and} \quad J \leftarrow (\Gamma(I) - \Gamma(B_c - I)).$$

  All edges from goods in $G_c - J$ to buyers in I are removed. Once again, this is justified in Lemma 15.

Assume that $(j,i), \; j \in J, \; i \in (B_c - I)$ is the new edge added to S_i in the current iteration. Observe that if $I' = \emptyset$, then all the flow from j, which was going to buyers in I before the addition of this edge, must go to i, since there is no residual path from I to i. Accordingly, **Update sets(I)** will move good j from J to $G_c - J$. As soon as the prices of goods in J are reduced by an infinitesimally small amount (by decreasing x) buyers in I will not be interested in good j anymore.

Lemma 11 *In Stage I, at the start of each iteration, for each buyer $i \in I$ there is a good $j \in J$ such that edge (j,i) is in the network.*

Proof : Since $\beta_i < 0$, the balanced flow must be sending flow on some edge (j,i). If an edge (j,i') to a buyer $i' \in (B_c - I)$ is also present in the network, then there will be a residual path from i to i', violating Property 1. Therefore, there is no such edge and $j \in (\Gamma(I) - \Gamma(B_c - I))$, proving the lemma. □

We now explain the purpose of the sets B' and G'. Once a good is moved into G', its price gets frozen until the end of Stage I. At any point in Stage I, these sets satisfy the following properties:

1. W.r.t. the frozen prices of goods in G', for each buyer $i \in B'$, $\beta_i < 0$.
2. Buyers in B are totally uninterested in goods in G' at any price, i.e., for every $i \in B$ and every $j \in G'$, $u_{ij} = 0$. Hence, as prices of goods in G' are decreased, no edge from B to G' will ever enter the network.

The reason for the name "adaptable" is that as far as determining feasibility or infeasibility goes, buyers in B' can be made consistent with the outcome of the remaining buyers. Thus, if $\forall i \in B_c,\ \beta_i < 0$, by assigning the frozen goods in G' their prices at the time of freezing, we can ensure that $\forall i \in B,\ \beta_i < 0$. The resulting price vector is clearly feasible. In Step 9 in Algorithm 1 we have refer to this process as "restoring prices of adjustable goods."

If on the other hand $\sum_{i \in B_c} \beta_i \geq 0$, then we can give a proof of infeasibility in one of two ways. First, by lowering the prices of all goods in G' to zero, which can be done without introducing any new edges in the network, we can ensure that $\forall i \in B',\ \beta_i = 0$, thereby ensuring that these buyers don't affect the sum of β_i's. The conditions of Lemma 8 now hold and yield a proof of infeasibility. Second, by assigning the frozen goods in G' their prices at the time of freezing, we can ensure that the prices of all goods are positive and the conditions of Lemma 9 now hold to yield a different proof of infeasibility.

At the start of a phase, the set $I \subseteq B_c$ of buyers having the smallest β values is identified. The goods they desire are put in set J. If at any point, I and J are found to be adaptable, the algorithm updates B' and G' and the phase comes to an end. Otherwise, the algorithm lowers the prices of goods in J until a new edge (j, i), with $j \in J$ and $i \in (B_c - I)$ is added to the network. On recomputing a balanced flow, either β_i becomes negative, if so i moves into I and the iteration comes to an end, or for some buyer(s) $i' \in I$, $\beta_{i'}$ increases. If for some buyer $i \in I$, β_i becomes non-negative, the phase comes to an end. Lemma 12 shows that eventually, either all buyers are rendered good or the conditions of Lemma 8 start holding. In the former case, the frozen prices of goods in G' are restored and the algorithm moves on to Stage II to find equilibrium prices.

One way to view the operation of Stage I is as a tug-of-war between two sets of buyers: the good buyers and the rest. The algorithm decreases the prices of goods desired by buyers in I, thereby increasing their β_i's. This helps towards reaching the infeasibility condition stated above. However, as new edges enter the network and a balanced flow is recomputed, buyers may move between the 2 sets.

Observe that in each iteration, the algorithm needs to compute the largest value of x at which a new edge is added to the network. For any one edge this is straightforward; taking the maximum over all relevant edges gives the required value.

Lemma 12 *Stage I must terminate with either a feasible price vector or a proof of infeasibility.*

Proof : By Lemma 25, Stage I must terminate. If at this point, $\forall i \in B :\ \beta_i < 0$, a feasible price vector has been found. Otherwise, $\sum_{i \in B_c} \beta_i \geq 0$ and hence $\exists i \in B_c :\ \beta_i \geq 0$. This buyer must be in $B_c - I$, since all buyers in I satisfy $\beta_i < 0$, and the goods this buyer desires must have positive prices. Now, the conditions of Lemma 8 can be made to hold by setting the prices of goods in G' to zero. □

8.2 Details of Stage II

Algorithm 2 gives the pseudo code for Stage II. In this section, we give the subroutines used by this stage and Section 8.3 gives formal definitions of the predicates used in the While loops. If Stage I terminates with a feasible price vector $\boldsymbol{p}$, the algorithm moves to Stage II to find equilibrium prices. Since $\boldsymbol{p}$ is small, Stage II needs to raise prices of goods to get to the equilibrium. Another way to view the situation is that since buyers have surplus money, Stage II needs to change prices in such a way that the surplus drops to zero. Are both these requirements compatible, i.e., will the surplus of buyers decrease by raising prices? The following lemma clarifies this crucial point in the simplified setting of Lemma 10, i.e., prices of all goods are raised; of course, Stage II will raise the prices of well-chosen subsets of G.

Lemma 13 *If in the setting of Lemma 10, prices $\boldsymbol{p}$ are feasible and x is raised without violating the Invariant, then the surplus of each buyer decreases and the resulting price vector is still feasible.*

Proof : Since $\boldsymbol{p}$ is feasible, for each buyer i, $\beta_i < 0$. Clearly, if $x > 1$, $x \cdot \beta_i < \beta_i$, i.e. the surplus of buyer i decreases. Moreover, the property that the β of each buyer is negative is preserved. Hence the resulting price vector is still feasible. □

A run of Stage II is also partitioned into *phases*, which are further partitioned into *iterations*. An iteration ends when a new edge is added to the network and a phase ends when a new set goes tight. We will say that $S \subseteq G$ is a *tight set* if the total price of goods in S exactly equals the money possessed by buyers who are interested in goods in S, i.e., $p(S) = m(\Gamma(S))$. Clearly, if S is tight, buyers in $\Gamma(S)$ must have zero surplus and hence have $\beta_i = -1$. In each iteration, the algorithm computes a balanced flow in the current network, $N(\boldsymbol{p})$.

The subroutines used in Stage II are:

- **Find sets(II):** Sets $I \subseteq B$ and $J \subseteq G$ are initialized as follows.

  $$I \leftarrow \arg\max_{i \in B}\{\theta_i\} \quad \text{and} \quad J \leftarrow \Gamma(I).$$

  All edges are removed from goods in J to buyers in $B - I$; this is justified in Lemma 15 below.

- **Update sets(II):** Find the set, I', of all buyers in $B - I$ that have residual paths to buyers in I. Update

  $$I \leftarrow (I \cup I') \quad \text{and} \quad J \leftarrow \Gamma(I).$$

All edges are removed from goods in J to buyers in $B - I$. Once again, this is justified in Lemma 15.

Observe that if (j, i) is the new edge added to S_i, then good j must move from $G - J$ to J, whether or not $I' = \emptyset$. The choice of set J above ensures that if the prices of goods in J are increased by an infinitesimally small amount (by increasing x as stated in Algorithm 2), there is no change in the maximum bang-per-buck goods of buyers in B.

Lemma 14 *In Stage II, at the start of each iteration, for each buyer $i \in (B - I)$ there is a good $j \in (G - J)$ such that edge (j, i) is in the network.*

Proof : Since $\theta_i < 1$, the balanced flow must be sending flow on some edge (j, i). If $j \in J$, then there will be a residual path from i to a buyer in I, violating Property 1. Therefore, $j \in (G - J)$. □

Lemma 15 *In Stage I (Stage II), the Invariant holds after all edges from goods in $G - J$ (J) to buyers in I $(B - I)$ are removed.*

Proof : The idea of the proof is the same for both statements. In Stage I, right after **Update sets(I)** is executed, there are no residual paths from I to $B - I$. Therefore, by Property 1, any edges from $G - J$ to I could not be carrying any flow and hence their removal will not affect the Invariant.

In Stage II, right after **Update sets(II)** is executed, there are no residual paths from $B - I$ to I. Therefore, by Property 1, any edges from J to $B - I$ could not be carrying any flow and hence their removal will not affect the Invariant. □

In each iteration, we need to compute the smallest value of x at which a new edge is added to the network or a new set goes tight. The former computation is the same as in Stage I. Let the smallest value of x at which a new set goes tight be x^*. Let $b = \min_{i \in I} \left\{ -\frac{1}{\beta_i} \right\}$. Clearly, $b > 1$. Using Lemma 10, proved in Section 7, for x in the range $1 \leq x \leq b$, we prove below that $x^* = b$.

Lemma 16 *$x^* = b$.*

Proof : By definition of b, for $1 \leq x < b$, for each $i \in I$, the surplus of i will be $1 + x\beta_i > 0$, since $x\beta_i > -1$. Thus, each edge (i, t) will have positive surplus, implying that there are no tight sets.

Next, assume that $x = b$. Let

$$T = \left\{ i \in I \mid -\frac{1}{\beta_i} = b \right\} \quad \text{and} \quad S = \{ j \in J \mid f(j, i) > 0, \quad \text{for some} \quad i \in T \}.$$

Since f is a balanced flow in $N(\boldsymbol{p})$, there cannot be an edge (j, i) for $j \in S$ and $i \in (I - T)$ in $N(\boldsymbol{p})$. This is so because otherwise there would be a path from T to i in the residual graph, contradicting Property 1 (observe that $\beta_i < -1/b$). Therefore, $\Gamma(S) = T$. Moreover, for $i \in T$, the surplus on edge (i, t) w.r.t. flow $x \cdot f$ in $N(x\boldsymbol{p})$ is $1 + x(-1/b) = 0$. Hence S is a tight set in network $N(x\boldsymbol{p})$ for $x = b$. □

8.3 Predicates used in While loops

In Step 2 (Stage I), "a proof of feasibility is reached" when $\forall i \in B,\ \beta_i < 0$, and "a proof of infeasibility is reached" when $\sum_{i \in B} \beta_i \geq 0$. Thus "a proof of feasibility or infeasibility is not reached" is satisfied iff $\neg((\sum_{i \in B} \beta_i \geq 0) \vee (\forall i \in B,\ \beta_i < 0))$.

In Step 4 (Stage I), "$B - I$ desire J" is satisfied iff $\exists i \in (B - I),\ \exists j \in J : \quad u_{ij} > 0$, and "buyers in I have small surplus" is satisfied iff $\forall i \in I,\ \beta_i < 0$.

In Step 1, "a buyer in B has surplus money" is satisfied iff $\exists i \in B,\ \theta_i > 0)$.

In Step 3, "no set in J is tight" is satisfied iff $\neg(\exists S,\ \emptyset \subset S \subseteq J\ s.t.\ S$ is tight$)$.

8.4 The role of balanced flow

Besides being used for defining the central notion of feasible prices, balanced flow plays the following three, rather diverse, crucial roles in both stages of our algorithm.

1. Ensure that edges, that need to be removed as prices of goods in J are raised, did not carry any flow and hence their removal would not violate the Invariant; this is argued in Lemma 15

2. Ensure that in each iteration, buyers entering I in Stage I (Stage II) have sufficiently large $|\beta_i|$ ($|\theta_i|$); this is established in Lemma 21 (Lemma 31).

3. Prove that sufficient progress is made in an iteration and hence in a phase. This is established in Lemma 22 for Stage I and Lemma 32 for Stage II.

As stated in the Introduction, balanced flow could have been defined without resorting to the l_2 norm – as a max-flow that makes the surplus vector lexicographically smallest, after its components are sorted in decreasing order (and hence making the components as balanced as possible). It is easy to prove Property 1 with this definition as well. The first two roles listed above make use of Property 1 only. On the other hand, the third role uses the definition of balanced flow via the l_2 norm and as argued in Section 11, the use of the l_2 norm seems indispensable.

Algorithm 1 (Initialization and Stage I of the Algorithm for ADNB)

```
1. Initialization:
   (i) $\forall i \in B: \quad m_i \leftarrow 1$.
   (ii) Use the DPSV algorithm to compute equilibrium prices, $\boldsymbol{p}$.
   (iii) $\forall i \in B: \quad m_i \leftarrow 1 + \frac{c_i}{\gamma_i}$.
   (iv) $B_c \leftarrow B; \qquad G_c \leftarrow G$.
   (v) $B' \leftarrow \emptyset; \qquad G' \leftarrow \emptyset$.
   (vi) Compute a balanced flow in $N(\boldsymbol{p})$.

                    Stage I

2. (New Phase) While a proof of feasibility or infeasibility is not reached do:
3. Find sets(I).
4. (New Iteration) While $B_c - I$ desire $J$ and buyers in $I$ have small surplus do:
5.  Multiply the prices of goods in $J$ and $\alpha$'s of buyers in $I$ by $x$.
    Initialize $x \leftarrow 1$, and decrease $x$ continuously until:
    A new edge $(j, i)$ enters $S_i$, for $j \in J$ and $i \in (B_c - I)$.
        Add $(j, i)$ to $N(\boldsymbol{p})$ and compute a balanced flow in it.
        Update sets(I).
6. End (End Iteration)
7. If $\forall i \in (B_c - I),\ \forall j \in J: \quad u_{ij} = 0$, then:
    Declare $I$ and $J$ adaptable, i.e,,
    $G' \leftarrow (G' \cup J)$ and $G_c \leftarrow (G_c - J)$.
    $B' \leftarrow (B' \cup I)$ and $B_c \leftarrow (B_c - I)$.
8. End (End Phase)
9. If $\forall i \in B_c,\ \beta_i < 0$, then:
    Restore prices of adaptable goods in $G'$.
    Compute a balanced flow in $N(\boldsymbol{p})$.
    Go to Step 1 in Stage II.
10. Else (i.e., $\sum_{i \in B_c} \beta_i \geq 0$), output "The game is infeasible".
    HALT.
```

Algorithm 2 (Stage II of the Algorithm for ADNB)

```
1. (New Phase) While a buyer in $B$ has surplus money do:
2. Find sets(II).
3. (New Iteration) While no set in $J$ is tight do:
4.  Multiply prices of goods in $J$ and $\alpha$'s of buyers in $I$ by $x$.
    Initialize $x \leftarrow 1$, and raise $x$ continuously until:
    A new edge $(j, i)$ enters $S_i$, for $j \in (G - J)$ and $i \in I$.
        If so, add $(j, i)$ to $N(\boldsymbol{p})$ and compute a balanced flow in it.
        Update sets(II).
5. End (End Iteration)
6. End (End Phase)
7. Output the current allocations and prices.
   HALT.
```

9 Running Time Analysis

We first define some parameters of the given problem instance. Recall that $g = |G|$ and $n = |B|$. Let $U = \max_{i \in B, j \in G}\{u_{ij}\}$, $C = \max_{i \in B} c_i$, and $\Delta = nCU^n$. Observe that program (2) with all $c_i = 0$ is the same as the convex program for a linear Fisher market with all buyers having unit money. Hence, Theorem 2 gives a lower bound on the price of a good computed in Initialization. Let this lower bound be $1/\mu$, $\mu \in \mathbf{Z}^+$.

The following enhanced version of Lemma 10 will be needed in both stages.

Lemma 17 *Let f be a balanced flow in network $N(\boldsymbol{p})$. Then, for $0 < x \leq b$, the flow $x \cdot f$ is a balanced flow in $N(x\boldsymbol{p})$.*

Proof : For $i, j \in B$ assume that $1 + x\beta_i < 1 + x\beta_j$. Since $x > 0$, $1 + \beta_i < 1 + \beta_j$, i.e., w.r.t. flow f in $N(\boldsymbol{p})$, the surplus of i is smaller than that of j. Since f is a balanced flow in $N(\boldsymbol{p})$, by Property 1, there is no path from i to j in the residual graph. Therefore, w.r.t. flow $x \cdot f$ in $N(x\boldsymbol{p})$ also there is no path from i to j in the residual graph. Therefore, flow $x \cdot f$ in $N(x\boldsymbol{p})$ satisfies Property 1 and hence is a balanced flow. □

9.1 Stage I

Throughout Stage I, we will consider a partitioning of B_c into two sets, B_1 and B_2, containing buyers having $\beta_i < 0$ and $\beta_i \geq 0$, respectively. For Stage I, we will work with the following potential function:

$$\Phi = \sum_{i \in B_1} \beta_i^2.$$

As Stage I proceeds, buyers move from B_c to B', and within B_c between the sets B_1 and B_2. For this reason, it will be convenient to define Φ using an n-dimensional vector, ψ, called the *associated vector* of network N. The i-th component of this vector, ψ_i, is β_i for $i \in B_1$, and is 0 for $i \in (B' \cup B_2)$. Hence, an alternative definition for the potential function is:

$$\Phi = \|\psi\|^2.$$

Lemma 18 *In Stage I, a phase consists of at most ng iterations.*

Proof : Observe that if in **Update sets(I)**, $I' = \emptyset$, then a good must move from J to $G_c - J$. Otherwise, a buyer must move from $B_c - I$ to I. Clearly, there can be at most $|J| < |G_c|$ contiguous iterations of the first type and a total of at most $|B_c - I_0| < |B_c|$ iterations of the second type, where I_0 is the set I at the start of the phase. □

The central fact established below is that Φ drops by a factor of $(1 - 1/(gn^2))$ in a phase (Lemma 23). Towards this end, assume that a given phase consists of k iterations. Let I_0 denote set I at the start of the phase and let I_l denote the set I at the end of the l-th iteration, $1 \le l \le k$. Assume that at the start of this phase, $\max_{i \in B_1}\{|\beta_i|\} = \delta = \delta_0$. Let

$$\delta_l = \min_{i \in I_l}\{|\beta_i|\}, \quad \text{for} \quad 1 \le l < k, \quad \text{and} \quad \delta_k = 0.$$

As we will see in this section, the potential function Φ drops monotonically in each iteration in the phase. Within an iteration, we will account for the drop in two steps. First, as prices of goods in J are reduced, by Lemma 10 the β_i's of buyers $i \in I$ increase, leading to a reduction in Φ. Second, when a new edge (j, i), with $j \in J$ and $i \in (B_c - I)$, is added to the network, the flow becomes more balanced, leading to a further drop. We will account for these two reductions separately, via different arguments (see Lemma 22). For the first step, we work with the l_1 norm, establishing an increase in $\sum_{i \in B_1} \beta_i$. In the second step, $\sum_{i \in B_1} \beta_i$ will not change if $i \in (B_1 - I)$. Instead, we establish a decrease in $\|\psi\|^2$ using an l_2 norm based argument. We observe that the latter argument is difficult to apply to the first step since the money of buyers changes as prices change. Also, we do not know of a simple one step argument that accounts for the entire reduction in an iteration.

Next, we prove a key fact that accounts for the second decrease. Just before new edge (j, i) is added to S_i, let N be the network and $\boldsymbol{p}$ be the prices of goods. Let N' be the network obtained by adding this edge to N; of course, the prices remain unchanged. Let f and f^* be balanced flows in N and N', respectively, and let ψ_i and ψ_i^* be their associated vectors.

Lemma 19 $\|\psi\|^2 - \|\psi^*\|^2 \ge \sum_{h \in B_1} (\psi_h - \psi_h^*)^2$.

Proof : Since the Invariant holds and the prices are unchanged, f and f^* have the same value. Therefore, flow $f^* - f$ will consist of circulations. Since f is a balanced flow, all these circulations must use the edge (j, i), because otherwise a circulation not using edge (j, i) could be used for making f more balanced. These circulations will have the effect of increasing the surplus of certain buyers in I, say i_l, for $1 \leq l \leq k$, and decreasing the surplus of buyer $i \in (B_c - I)$. Let $\beta_i - \beta_i^* = \delta$, and for $1 \leq l \leq k$, $\beta_{i_l}^* - \beta_{i_l} = \delta_l$. Then, $\sum_{l=1}^{k} \delta_l = \delta$.

For each buyer i_l, $1 \leq i_l \leq k$, there is a path from i_l to i in the corresponding circulation and hence there is a path from i to i_l in the residual graph w.r.t. flow f^*. Since f^* is balanced, by Property 1, the surplus of buyer i is at least as large as that of i_l. Therefore, $\beta_i^* \geq \beta_{i_l}^*$.

In going from N to N', the ψ_h values can change only for $h = i$, and $h = i_l$, for $1 \leq l \leq k$. We will consider 3 cases.

Case 1: $\beta_i \geq 0$, i.e., $i \in B_2$, and $\beta_i^* \geq 0$.
In this case, $\psi_i = \psi_i^* = 0$ and the lemma is obvious.

Case 2: $\beta_i \geq 0$, i.e., $i \in B_2$, and $\beta_i^* < 0$.
Let $a = -\beta_i^*$. In this case, $\psi_i = 0$ and $\psi_i^* = -a$.

Clearly, $a \leq \sum_{l=1}^{k} \delta_l$ Since $\beta_i^* \geq \beta_{i_l}^*$, $a \leq b_l - \delta_l$. Now,

$$\begin{aligned}\|\psi\|^2 - \|\psi^*\|^2 &= \left(0^2 + \sum_{l=1}^{k} b_l^2\right) - \left(a^2 + \sum_{l=1}^{k} (b_l - \delta_l)^2\right) = -a^2 + \sum_{l=1}^{k} (2b_l - \delta_l)\delta_l \\ &\geq -a^2 + \sum_{l=1}^{k} (2a + \delta_l)\delta_l \geq -a^2 + \sum_{l=1}^{k} \delta_l^2 + 2a \sum_{l=1}^{k} \delta_l \geq \sum_{l=1}^{k} \delta_l^2,\end{aligned}$$

where the first inequality follows from $a \leq b_l - \delta_l$ and the third one follows from $a \leq \sum_{l=1}^{k} \delta_l$.

Case 3: $\beta_i^* < 0$, i.e., $i \in (B_1 - I)$.
Clearly, in this case, $\beta_i < 0$. Substitute $a = -\beta_i$, and for $1 \leq i_l \leq k$, substitute $b_l = -\beta_{i_l}$. Now, by Lemma 20, to get $\|\psi\|^2 - \|\psi^*\|^2 \geq \delta^2$. Clearly, $\delta^2 \geq \sum_{l=1}^{k} \delta_l^2$, giving the lemma. □

Lemma 20 *Let $\delta, \delta_l \geq 0$, $l = 1, 2, \ldots, k$, with $\delta = \sum_{l=1}^{k} \delta_l$. If $a + \delta \leq b_l - \delta_l$, for $l = 1, 2, \ldots, k$ then*

$$\|(a, b_1, b_2, \ldots, b_k)\|^2 - \|(a + \delta, b_1 - \delta_1, b_2 - \delta_2, \ldots, b_k - \delta_k)\|^2 \geq \delta^2.$$

Proof :

$$\begin{aligned}&\left(a^2 + \sum_{l=1}^{k} {b_l}^2\right) - \left((a + \delta)^2 + \sum_{l=1}^{k} (b_l - \delta_l)^2\right) \\ &\geq \left((a + \delta - \delta)^2 + \sum_{l=1}^{k} (b_l - \delta_l + \delta_l)^2\right) - \left((a + \delta)^2 + \sum_{l=1}^{k} (b_l - \delta_l)^2\right) \\ &\geq \delta^2 + 2(a + \delta)\left((\sum_{l=1}^{k} \delta_l) - \delta\right) \quad \geq \quad \delta^2.\end{aligned}$$

□

Let ψ^0 denote vector ψ at the start of the phase and ψ^l denote ψ at the end of iteration l, for $1 \leq l \leq k$.

Lemma 21 *In the l-th iteration, there is a buyer $i \in I_{l-1}$ such that $|\beta_i|$ decreases by at least $(\delta_{l-1} - \delta_l)$, for $1 \leq l < k$.*

Proof : By the definition of set I' in procedure **Update sets(I)** and Property 1, there is a buyer $i \in I_{l-1}$ which achieves $\min_{i \in I_l}\{|\beta_i|\}$ at the end of iteration l. Clearly, β_i increases (and hence $|\beta_i|$ decreases) by at least $(\delta_{l-1} - \delta_l)$ in the l-th iteration. □

Lemma 22 *For $1 \leq l \leq k$,*

$$\|\psi^{l-1}\|^2 - \|\psi^l\|^2 \geq (\delta_{l-1} - \delta_l)^2.$$

Proof : We first prove the statement for $1 \leq l < k$. By Lemma 21, there is a buyer $i \in I_{l-1}$ such that β_i increases by at least $(\delta_{l-1} - \delta_l)$ in the l-th iteration. Let us split this increase into two parts, the increase due to decrease in the prices of goods in J and that due to a new edge entering the network. Let these be a and b, respectively. Therefore, $a + b = \delta_{l-1} - \delta_l$.

Let ψ' be the vector ψ just before the new edge is added to the network in iteration l, i.e., right after all the decrease in prices of J has happened. As prices in J decrease, the beta's of buyers in I increase, each leading to a decrease in $\|\psi'\|^2$; clearly, the beta's of buyers in $B_c - I$ remain unchanged. Let c be the value of beta of buyer i at the beginning of iteration l. Then,

$$\|\psi^{l-1}\|^2 - \|\psi'\|^2 \geq c^2 - (c + a)^2 = a^2 - 2ac.$$

By Lemma 19,

$$\|\psi'\|^2 - \|\psi^l\|^2 \geq b^2.$$

Adding the two we get

$$\|\psi^{l-1}\|^2 - \|\psi^l\|^2 \geq a^2 - 2ac + b^2 \geq (\delta' + \delta'')^2 \geq (\delta_{l-1} - \delta_l)^2,$$

where the second last inequality follows from the observation that $b \leq -c$.

Finally, in the k-th iteration, there is a buyer $i \in I_{k-1}$ whose ψ_i changes from $\beta_i < 0$ to 0. Therefore,

$$\|\psi^{k-1}\|^2 - \|\psi^k\|^2 \geq \beta_i^2 \geq (\delta_{k-1} - \delta_k)^2,$$

since $\delta_{k-1} \leq -\beta_i$ and $\delta_k = 0$. □

Lemma 23 *In a phase in Stage I, the potential drops by a factor of*

$$\left(1 - \frac{1}{n^2 g}\right).$$

Proof : Now, $\|\psi^0\|^2 - \|\psi^k\|^2$ can be written as a telescoping sum of k terms, each of which is the decrease in the potential in one of the k iterations. Lemma 22 gives a lower bound on each of these terms. The total lower bound is minimized when each of the differences $(\delta_{l-1} - \delta_l)$ is equal. Now using the fact that $\delta_0 = \delta$ and $\delta_k = 0$, we get:

$$\|\psi^0\|^2 - \|\psi^k\|^2 \geq \frac{\delta^2}{k}.$$

Finally, since $\|\psi^0\|^2 \leq n\delta^2$, and by Lemma 18 $k \leq ng$, we get:

$$\|\psi^k\|^2 \leq \|\psi^0\|^2 \left(1 - \frac{1}{n^2 g}\right).$$

□

Lemma 24 *At any point in Stage I, if* $\sum_{i \in B_2} \beta_i > 0$*, then*

$$\sum_{i \in B_2} \beta_i \geq \frac{1}{U^n \mu^g}.$$

Proof : First observe that the maximum bang-per-buck of buyers in B_2 remains unchanged throughout Stage I, and is determined by prices of goods found in the Initialization. Let $J_2 = \Gamma(B_2)$. By Property 1, there is no flow from a good in J_2 to a buyer in B_1, and therefore, all flow from J_2 must go to buyers in B_2. Therefore,

$$\sum_{i \in B_2} \beta_i = \sum_{i \in B_2} \theta_i - |B_2| = \sum_{i \in B_2} \alpha_i - \sum_{j \in J_2} p_j.$$

Now if $\sum_{i \in B_2} \beta_i > 0$, then the denominator of this sum is a product of at most n u_{ij}'s and at most g p_j's, and is therefore bounded by $U^n \mu^g$, proving the lemma. □

Lemma 25 *The execution of Stage I requires at most*

$$O\left(n^4 g^2 (n \log U + g \log \mu)\right)$$

max-flow computations.

Proof : By Lemma 23, the square of the potential drops by a factor of two after $O(n^2 g)$ phases. At the start of the algorithm, the potential is at most n.

If at any point, $\sum_{i \in B_2} \beta_i = 0$, i.e., $\forall i \in B_2 : \ \beta_i = 0$, each subsequent phase must end up in Step 5 and not Step 6, i.e., some buyers will be removed from consideration. Therefore, from this point on, at most n more phases are needed for Stage I to terminate.

Next, let's assume that $\sum_{i \in B_2} \beta_i > 0$ throughout Stage I. If so, by Lemma 24, once the potential drops below $\frac{1}{(U^n \mu^g)^2}$, the phase must end (in Step 7). Therefore the number of phases is

$$O(n^2 g \log(U^{2n} \mu^{2g})).$$

By Lemma 18 each phase consists of at most ng iterations and each iteration requires n max-flow computations for finding a balanced flow. The lemma follows. □

9.2 Stage II

When $\forall i \in B : \ \beta_i < 0$, the algorithm starts with Stage II. Since in this stage the algorithm only raises prices of goods (i.e., increases x), by Lemma 13, $\forall i \in B : \ \beta_i < 0$ holds until termination.

In this section, we will work with the θ_i's of buyers, rather than their β_i's. Thus, throughout Stage II, $\forall i \in B : \ \theta_i < 1$. For Stage II, we will work with the following potential function:

$$\Phi = \sum_{i \in B} \theta_i^2.$$

Lemma 26 *In Stage II, at the termination of a phase, the prices of goods in the newly tight set must be rational numbers with denominator $\leq \Delta$.*

Proof : Let S be the newly tight set. Consider the subgraph of the network induced on the bipartition $(S, \Gamma(S))$, and view this as an undirected graph, say H. Assume w.l.o.g. that this graph is connected (otherwise we prove the lemma for each connected component of H). Pick a spanning tree in H.

Pick any good $j \in S$, and find a path in the spanning tree from j to each good $j' \in S$. If j reaches j' with a path of length $2l$, then $p_{j'} = ap_j/b$ where a and b are products of l utility parameters (u_{ik}'s) each. Since alternate edges of this path contribute to a and b, we can partition the u_{ik}'s of edges in the spanning tree into two sets, T_1 and T_2, such that a uses u_{ik}'s from T_1 and b uses those from T_2.

Next, consider $\alpha_i = c_i/\gamma_i$, for $i \in \Gamma(S)$. Now, $\gamma_i = u_{i,j'}/p_{j'}$, where (i, j') is any edge in the network. Find the path in the spanning tree from i to j and use the first edge on this path for computing γ_i (it is easy to see that all these edges come from set T_2), and substitute $p_{j'}$ using the expression stated above, i.e., $p_{j'} = ap_j/b$.

Since S is a tight set,

$$\sum_{j' \in S} p_{j'} = \sum_{i \in \Gamma(S)} 1 + \alpha_i.$$

In this equation, substitute for $p_{j'}$ and α_i using the expressions constructed above to get an equation with one variable, i.e., p_j. Now, it is easy to see that the denominator of p_j is $\leq \Delta$. □

Lemma 27 *In Stage II, consider two phases P and P', not necessarily consecutive, such that good j lies in the newly tight sets at the end of P as well as P'. Then the increase in the price of j, going from P to P', is at least $1/\Delta^2$.*

Proof : Let the prices of j at the end of P and P' be p/q and r/s, respectively. Clearly, $r/s > p/q$. By Lemma 26, $q \leq \Delta$ and $r \leq \Delta$. Therefore the increase in price of j,

$$\frac{r}{s} - \frac{p}{q} \geq \frac{1}{\Delta^2}.$$

□

Lemma 28 *In Stage II, a phase consists of at most g iterations.*

Proof : After each iteration, other than the last one, at least one good will move from $G-J$ to J. □

The structure of the rest of the argument is quite similar to that of Stage I. Once again, the central fact established is that Φ drops by an inverse polynomial factor, of $(1 - 1/n^2)$, in a phase (Lemma 33). Assume that a given phase consists of k iterations. Let I_0 denote set I at the start of the phase and let I_l denote the set I at the end of the l-th iteration, $1 \leq l \leq k$. Assume that at the start of this phase, $\max_{i \in B}\{|\theta_i|\} = \delta = \delta_0$. Let

$$\delta_l = \min_{i \in I_l}\{|\theta_i|\}, \quad \text{for} \quad 1 \leq l < k, \quad \text{and} \quad \theta_k = 0.$$

As in Stage I, we will account for the drop in Φ in two steps in each iteration. First, as prices of goods in J are increased, the θ_i's of buyers $i \in I$ decrease, leading to a reduction in Φ. Second, when a new edge (j, i), with $j \in (G_c - J)$ and $i \in I$, is added to the network, the flow becomes more balanced, leading to a further drop. As in Stage I, we will account for the first drop using the l_1 norm and the second drop using the l_2 norm.

We begin by accounting for the second decrease. Just before new edge (j, i) is added, let N be the network and $\boldsymbol{p}$ be the prices of goods. Let N' be the network obtained by adding this edge to N; of course, the prices remain unchanged. Let f and f^* be balanced flows in N and N', respectively. Denote by θ (θ^*) the surplus vector w.r.t. flow f in N (flow f^* in N').

Lemma 29 $\|\theta\|^2 - \|\theta^*\|^2 \geq \delta^2$, *where $\delta = \theta_i - \theta_i^*$.*

Proof : Since the Invariant holds and the prices are unchanged, f and f^* have the same value. Therefore, flow $f^* - f$ will consist of circulations. Since f is a balanced flow, all these circulations must use the edge (j, i), because otherwise a circulation not using edge (j, i) could be used for making f more balanced. These circulations will have the effect of decreasing the surplus of buyer $i \in I$, and increasing the surplus of buyers $i_l \in (B - I)$, for $1 \leq l \leq k$. Let $\theta^*_{i_l} - \theta_{i_l} = \delta_l$, for $1 \leq l \leq k$. Then, $\sum_{l=1}^{k} \delta_l = \delta$.

For each buyer i_l, $1 \leq i_l \leq k$, there is a path from i_l to i in the corresponding circulation and hence there is a path from i to i_l in the residual graph w.r.t. flow f^*. Since f^* is balanced, by Property 1, the surplus of buyer i w.r.t. f^* is at least as large as that of i_l. Therefore, $\theta^*_i \geq \theta^*_{i_l}$. The inequality $\|\theta\|^2 - \|\theta^*\|^2 \geq \delta^2$ now follows from Lemma 30, on substituting $a = \theta^*_i$, and for $1 \leq l \leq k$, $b_l = \theta^*_{i_l}$. □

Lemma 30 *If $a \geq b_l \geq 0, l = 1, 2, \ldots, k$ and $\delta = \sum_{l=1}^{k} \delta_l$ where $\delta, \delta_l \geq 0, l = 1, 2, \ldots, k$, then*

$$\|(a + \delta, b_1 - \delta_1, b_2 - \delta_2, \ldots, b_k - \delta_k)\|^2 - \|(a, b_1, b_2, \ldots, b_k)\|^2 \geq \delta^2.$$

Proof :

$$(a + \delta)^2 + \sum_{i=1}^{k} (b_i - \delta_i)^2 - a^2 - \sum_{i=1}^{k} {b_i}^2 \geq \delta^2 + 2a(\delta - \sum_{i=1}^{k} \delta_i) \geq \delta^2.$$

□

Let θ^0 denote the surplus vector at the start of the phase and let θ^l denote the surplus vector at the end of iteration l, for $1 \leq l \leq k$.

Lemma 31 *In the l-th iteration, there is a buyer $i \in I_{l-1}$ whose surplus decreases by at least $(\delta_{l-1} - \delta_l)$, for $1 \leq l < k$.*

Proof : By the definition of set I' in procedure **Update sets(II)** and Property 1, there is a buyer $i \in I_{l-1}$ which achieves $\min_{i \in I_l}\{|\theta_i|\}$ at the end of iteration l. Clearly, the surplus of i decreases by at least $(\delta_{l-1} - \delta_l)$ in the l-th iteration. □

Lemma 32 *For $1 \leq l \leq k$,*

$$\|\theta^{l-1}\|^2 - \|\theta^l\|^2 \geq (\delta_{l-1} - \delta_l)^2.$$

Proof : We first prove the statement for $1 \leq l < k$. By Lemma 31, there is a buyer $i \in I_{l-1}$ whose surplus decreases by at least $(\delta_{l-1} - \delta_l)$ in the l-th iteration. Let us split this decrease into two parts, the decrease due to increase in the prices of goods in J and that due to a new edge entering the network. Let these be a and b, respectively. Clearly, $a + b \geq \delta_{l-1} - \delta_l$.

Let θ' be the surplus vector just before the new edge is added to the network in iteration l, i.e., right after all the increase in prices of J has happened. As prices in J increase, the surpluses of buyers in I decrease, but those of buyers in $B - I$ remain unchanged. Let c be the surplus of buyer i at the beginning of iteration l. Then,

$$\|\theta^{l-1}\|^2 - \|\theta'\|^2 \geq c^2 - (c - a)^2 \geq a^2 + 2ac.$$

By Lemma 29,

$$\|\theta'\|^2 - \|\theta^l\|^2 \geq b^2.$$

Adding the two we get

$$\|\theta^{l-1}\|^2 - \|\theta^l\|^2 \geq a^2 + 2ac + b^2 \geq (a+b)^2 \geq (\delta_{l-1} - \delta_l)^2,$$

where the second last inequality follows from the observation that $b \leq c$.

Finally, in the k-th iteration, there is a buyer $i \in I_{k-1}$ whose surplus changes from $\theta_i > 0$ to 0. Therefore,

$$\|\theta^{k-1}\|^2 - \|\theta^k\|^2 \geq \theta_i^2 \geq (\delta_{k-1} - \delta_k)^2,$$

since $\delta_{k-1} \leq \theta_i$ and $\delta_k = 0$. □

Lemma 33 *In a phase in Stage II, the potential drops by a factor of*

$$\left(1 - \frac{1}{n^2}\right).$$

Proof : Now, $\|\theta^0\|^2 - \|\theta^k\|^2$ can be written as a telescoping sum of k terms, each of which is the decrease in the potential in one of the k iterations. Lemma 32 gives a lower bound on each of these terms. The total lower bound is minimized when each of the differences $(\delta_{l-1} - \delta_l)$ is equal. Now using the fact that $\delta_0 = \delta$ and $\delta_k = 0$, we get:

$$\|\theta^0\|^2 - \|\theta^k\|^2 \geq \frac{\delta^2}{k}.$$

Finally, since $\|\theta^0\|^2 \leq n\delta^2$, and by Lemma 28 $k \leq n$, we get:

$$\|\theta^k\|^2 \leq \|\theta^0\|^2 \left(1 - \frac{1}{n^2}\right).$$

□

Lemma 34 *The execution of Stage II requires at most*

$$O\left(n^4(\log n + n\log U + \log C)\right)$$

max-flow computations.

Proof : By Lemma 33, the potential drops by a factor of half after $O(n^2)$ phases. At the start of the algorithm, the potential is at most n. Once its value drops below $1/\Delta^4$, the algorithm requires at most n more phases to compute equilibrium prices. This follows from Lemma 26 and Lemma 27. Therefore the number of phases is

$$O(n^2 \log(\Delta^4 n)) \;=\; O(n^2(\log n + n \log U + \log C)).$$

By Lemma 28 each phase consists of n iterations and each iteration requires n max-flow computations for computing a balanced flow. The lemma follows. □

Lemmas 25 and 34 give:

Theorem 35 *Algorithms 1 and 2 solve the decision and search versions of Nash bargaining game* **ADNB** *using*

$$O\left(n^4 g(\log n + n \log U + \log C + g \log \mu)\right)$$

max-flow computations.

10 Postmortem and Was Stage I Really Needed?

As pointed out in Section 4 in [DPSV08], the primal-dual paradigm operates in a fundamentally different way in the setting of a rational convex program than in the setting of an integral linear program. In the latter setting, in each iteration, it picks an unsatisfied complementary slackness condition and satisfies it. On the other hand, in the former setting, the algorithm starts off with a suboptimal solution that can be viewed as relaxing a class of the KKT conditions. It then tightens these conditions gradually; when they are all fully tightened, the optimal solution has been reached.

Let us first show that this high level picture applies to Stage II of our algorithm as well. Consider the situation right after a balanced flow has been computed at any point in Stage II, and consider an arbitrary buyer i. At this point, let f_i be the flow sent on edge (i,t) by the balanced flow; f_i is also the money spent by i in the current allocation. Buyer i's available money at this point is $m_i = 1 + c_i/\gamma_i$. Therefore,

$$f_i = m_i - \beta_i - 1 = \frac{c_i}{\gamma_i} - \beta_i.$$

Let v_i be the total utility derived by i from the current allocation and suppose that $x_{ij} > 0$. Then,

$$\gamma_i = \frac{u_{ij}}{p_j} = \frac{v_i}{f_i} = \frac{v_i}{(-\beta_i) + \frac{c_i}{\gamma_i}}.$$

This yields

$$\left((-\beta_i) + \frac{c_i}{\gamma_i}\right)\gamma_i = v_i \Rightarrow \gamma_i = \frac{v_i - c_i}{(-\beta_i)}.$$

Substituting for γ_i and rearranging we get

$$\frac{p_j}{(-\beta_i)} = \frac{u_{ij}}{v_i - c_i}.$$

To summarize, at any point in Stage II, we have ensured the first two KKT conditions and relaxed the last two as follows:

(1) $\forall j \in G: \ p_j \geq 0$.

(2) $\forall j \in G: \ p_j > 0 \Rightarrow \sum_{i \in B} x_{ij} = 1$.

(3') $\forall i \in B, \ \forall j \in G: \ \frac{p_j}{-\beta_i} \geq \frac{u_{ij}}{v_i - c_i}$.

(4') $\forall i \in B, \ \forall j \in G: x_{ij} > 0 \Rightarrow \frac{p_j}{-\beta_i} = \frac{u_{ij}}{v_i - c_i}$.

Observe that if prices are not feasible, then for some i, $\beta_i \geq 0$. If so, the relaxed KKT conditions (3') and (4') will be meaningless. Recall that throughout Stage II, $0 < -\beta_i \leq 1$ and $(-\beta_i)$ monotonically increases and reaches 1 at termination. Thus, at termination, the last two KKT conditions are also ensured. For establishing a bound on the number of phases needed in Stage II, it suffices to study the potential function

$$\Phi' = \sum_{i \in B} \beta_i^2.$$

Clearly, $\Phi' > 0$ at the start of Stage II and increases monotonically to n. However, it turns out to be more convenient to study the potential function Φ given in Section 9.2, which clearly achieves the same end.

Stage I determines feasibility of the given market. What if we only wanted to solve the promise problem of finding the equilibrium of a given feasible market? Since the prices found by Initialization are guaranteed to be small, could we not go directly to Stage II and raise prices until equilibrium is attained? The answer is "No". The reason is that Stage II is guaranteed to converge only if it is started with feasible prices. As can be seen from the proof of Lemma 10, if for some buyer i, $\beta_i > 0$, then raising x will actually increase her surplus. This happens because her money will increase at a faster rate than the rate at which flow on edge (i, t) increases.

Thus, Stage I not only determines feasibility but, if the given market is feasible, it also finds a suitable initial price vector for Stage II. An interesting aspect of Stage I is that it needs to decrease prices of goods, even though it starts with a small price vector. Furthermore, it is easy to see that if the given market is feasible and Stage I is started off with *any* small price vector, not necessarily the one found by Initialization, it will terminate with a feasible price vector. Hence we get the following interesting fact:

Lemma 36 *Let $\mathcal{M}$ be a feasible flexible budget market and let $\boldsymbol{p}$ be small, positive prices for it. Then, there exist positive prices $\boldsymbol{q}$ such that $\boldsymbol{p}$ weakly dominates $\boldsymbol{q}$ and prices $\boldsymbol{q}$ are feasible.*

Finally we address the question of whether a separate procedure was needed for testing feasibility, especially in light of the fact that such a procedure is not needed when the primal-dual paradigm is used for solving a non-total integral linear program. Let us illustrate the latter by comparing, at a high level, the algorithms for the problems of maximum weight matching and maximum weight perfect matching in bipartite graphs; for full details, see [CCPS98].

The only difference in the LP's of these two problems is that whereas the former demands at most 1 matched edge incident at each vertex and the latter demands exactly 1 edge. As a result, in the dual, the vertex variables are constrained to be non-negative in the former and unconstrained in the latter. This gives rise to an additional complementary slackness condition in the former, i.e., any vertex with a positive dual must be matched.

The algorithm for the former problem attempts to "repair" this complementary slackness condition one vertex at a time. It attempts to find an augmenting path from a violating vertex, say v, by growing a "Hungarian tree" rooted at v. If another unmatched vertex enters the tree, an augmenting path can be found and v is matched off. On the other hand, if the tree becomes maximal without encountering another unmatched vertex, then the algorithm is able to drive the dual of some vertex in the tree down to zero. If this vertex is v, the complementary slackness condition at v has been repaired. If it is some other vertex, say u, then there is an alternating path between u and v, which enables the algorithm to unmatch u and instead match off v. This repairs the complementary slackness condition at v without creating a violation at u.

The algorithm for the latter problem is very similar – it attempts to iteratively match off unmatched vertices. It tries to find an augmenting path from an unmatched vertex, say v, by growing a "Hungarian tree" rooted at v. If the tree becomes maximal without encountering another unmatched vertex, then the number of "outer" vertices in the tree exceeds the number of "inner" vertices by 1. Now, decreasing the duals at outer vertices by Δ and increasing them at inner vertices by Δ yields a feasible dual. By letting $\Delta \rightarrow \infty$, we get that the dual LP is unbounded. This gives a proof of infeasibility of the primal LP.

Consider the two problems **ADNB** and Fisher's linear case with the money of each agent being unit. Observe that the latter problem is a special case of the former when the disagreement utilities of all agents are zero, and the latter is total whereas the former is non-total. Why was the algorithm for the former so much more elaborate than that for the latter (i.e., the DPSV algorithm), especially in view of the fact that the algorithm for maximum weight perfect matching is not more involved than that for maximum weight matching?

In the case of perfect matching, feasibility was ensured one vertex at a time, and when it could not be ensured, we got a proof of infeasibility right away. In the case of rational convex programs, the KKT conditions were enforced gradually and these conditions were fully satisfied only right at the end. So, why can't we simply run Stage II right to the end and then obtain a solution or find out that the given instance was infeasible? The reason is that to guarantee termination of Stage II, we need to start it off with a feasible price vector, i.e., we need to determine feasibility before starting with Stage II. Hence Stage I appears to be essential.

11 l_1-norm Does Not Suffice

We give a family of examples showing that the DPSV algorithm, for Fisher's linear case, may end up making only inverse exponential progress in a phase if the potential function used is the l_1 norm of the surplus vector.

We will define the example in terms of 2 parameters, δ and H, which will be set at the end. Assume $B = \{b_0, b_1, \ldots, b_{n-1}, b_n\}$ and $G = \{g_0, g_1, \ldots, g_{n-1}, g_n\}$. At the start of the phase, the only edges present in the network are (g_i, b_i), for $0 \leq i \leq n$. The money of the buyers are as follows:

$$m_0 = 1 + \delta, \quad \text{and for } \ 1 \leq i \leq n-1, \ m_i = \frac{\delta}{2^i}, \ \text{ and } \ m_n = H + \frac{\delta}{n}.$$

The prices of goods are as follows:

$$p_0 = 1, \quad \text{and for } \ 1 \leq i \leq n-1, \ p_i = \frac{\delta}{2^i}, \ \text{ and } \ p_n = H + \frac{\delta}{n}.$$

Hence, at the start of the phase, the surplus of b_0 is δ, and that of the rest of the buyers is 0.

We will set $\delta = 1$ and H to be a large number, say n. The phase starts with $I = \{b_0\}$ and $J = \{g_0\}$. Assume that at the end of iteration i, edge (g_i, b_{i-1}) enters the network, and as a result, b_i enters I and g_i enters J, for $1 \leq i \leq n$. The increment in price in each iteration is very small – this is easily arranged by choosing the right utilities u_{ij}'s.

To keep the description clean, let us assume the increments in price are all zero; the numbers can be easily modified by inverse exponential amounts to yield the desired outcome, even if the prices need to increase in each iteration. If so, at the end of all this, the surplus of b_i is

$$\frac{\delta}{2^{i+1}}, \quad \text{for } \ 0 \leq i \leq n-1,$$

and that of b_n is $\frac{\delta}{2^{n-1}}$.

Finally, in iteration $n+1$, a very slight increase in x leads to set $\{g_n\}$ going tight. Observe that the reason for choosing H to be a large number is the ensure that this slight increase in x will not make a larger set go tight. Observe that $\Gamma(\{g_n\} = \{b_n, b_{n+1}\}$. Now, the increase in the price of g_n needed for this is $\frac{\delta}{2^{n-1}}$. Since H is a large number and the increase in x is very small, the total increase in the prices of other goods is at most a constant factor more.

In summary, the total increase in the l_1 norm of $\boldsymbol{p}$ in this phase is an inverse exponential factor.

12 Discussion

Our paper provides two new pieces of evidence to show that the notion of balanced flow is basic to the problems tackled in [DPSV08] and the current paper. The first is the result of Section 11. Second, observe that the notion of a feasible price vector was defined using balanced flows. In view of Lemma 6 and the remarks made after it, and of Lemma 36, characterizing the sets of small and feasible price vectors for a feasible flexible budget market is an interesting question.

As in the case of the EG-program, the optimal solutions of convex program (2) resemble those of a linear program rather than a nonlinear program. So, we repeat a question raised in [Vaz07] namely, can the solution to **ADNB** be captured via a linear program? We believe the answers to these questions are "no" and that establishing this in a suitable formal framework will provide new insights into the boundary between linear and nonlinear programs.

The most prominent problem for which a rational convex program is known but a combinatorial algorithm is not known is the linear case of the Arrow-Debreu market model, i.e., the convex program of Jain [Jai07]. We pose the following easier question: Give a polynomial time algorithm for this problem that uses only an LP solver. A much more general question along these lines is stated in the Introduction.

We list some more rational convex programs and leave the problem of finding combinatorial algorithms for them. First, generalize **ADNB** to additively separable, piecewise-linear, concave utilities. This problem has a rational convex program, and the question of finding a combinatorial algorithm for it becomes even more significant in view of recent results showing PPAD-completeness of computing an equilibrium in the Arrow-Debreu model with these utility functions [CDDT09, CT09, VY10].

In an interesting paper, Kalai [Kal77] relaxed Nash's axiom of symmetry and derived the solution concept of *nonsymmetric bargaining games*. The convex program capturing the solution to the nonsymmetric extension of **ADNB** is also rational; moreover, this convex program generalizes the Eisenberg-Gale program and hence captures Fisher's linear case as well. Despite substantial effort, this problem has not yielded to a combinatorial algorithm. Once it is obtained, one could consider the common generalization of the last two problems, i.e., nonsymmetric **ADNB** with additively separable, piecewise-linear, concave utilities.

On restricting **ADNB** (nonsymmetric **ADNB**) to zero disagreement utilities we get the problems of computing equilibria for linear Fisher markets with unit (arbitrary) money among buyers. Of course, both these problems are total. It turns out that a combinatorial algorithm for the unit money case is no easier than that for the arbitrary money case. In view of this, the difficulty of obtaining a combinatorial algorithm for nonsymmetric **ADNB** comes as a surprise and may be substantiating the observation that in the setting of rational convex programs, non-total problems behave quite differently from total problems.

All the rational convex programs mentioned above involve the log function in the objective, together with linear constraints. Another class of rational convex programs is obtained by having a quadratic objective function and linear constraints. It will be very interesting to obtain combinatorial polynomial time algorithms for such rational convex programs as well.

The reader can see that the algorithm for **ADNB** exploits a surprisingly rich and clean structure which is, in some ways, reminiscent of the majestic structure of matching. In our experience, such structure does not occur in isolation and we believe that what we see so far is the tip of an iceberg. This leads to the question: what does the rest of the iceberg look like?

One possibility is to seek combinatorial approximation algorithms for solving specific classes of nonlinear convex programs. In this respect, important hints may be obtained from the way the primal-dual paradigm was extended from solving linear programs exactly to obtaining near-optimal solutions to linear programs within the area of approximation algorithms. The mechanism involved in all of the latter algorithms was that of relaxing complementary slackness conditions, which was first formalized in [WGMV95]. We also note that in the setting of approximation algorithms, the primal-dual paradigm has been successful primarily for minimization problems. So, our more precise question is, "Is there a natural way of relaxing the KKT conditions to obtain primal-dual (combinatorial) algorithms for near-optimally solving interesting classes of (perhaps minimization) nonlinear convex programs?"

13 Acknowledgments

I am indebted to Ehud Kalai and Nimrod Megiddo for sharing their insights on the Nash bargaining game and starting me off on this research. I also wish to thank Matthew Andrews for several valuable discussions on the application of **ADNB** reported in Section 4.

References

[And09] M. Andrews. Personal communication. 2009.

[AQS05] M. Andrews, L. Qian, and A. Stolyar. Optimal uitlity based multi-user throughput allocation subject to throughput constraints. In *INFOCOM*, 2005.

[BDX10] B. Birnbaum, N. Devanur, and L. Xiao. New convex programs and distributed algorithms for Fisher markets with linear and spending constraint utilities. Unpublished manuscript, 2010.

[BS00] W. C. Brainard and H. E. Scarf. How to compute equilibrium prices in 1891. *Cowles Foundation Discussion Paper*, (1270), 2000.

[CCPS98] W.J. Cook, W.H. Cunningham, W.R. Pulleyblank, and A. Schrijver. *Combinatorial Optimization*. John Wiley & Sons, New York, NY, 1998.

[CDDT09] X. Chen, D. Dai, Y. Du, and S.-H. Teng. Settling the complexity of arrow-debreu equilibria in markets with additively separable utilities. In *FOCS*, 2009.

[CDV] D. Chakrabarty, N. Devanur, and V. V. Vazirani. New results on rationality and strongly polynomial solvability in Eisenberg-Gale markets. *SIDMA*. To appear.

[CT09] X. Chen and S.-H. Teng. Spending is not easier than trading: on the computational equivalence of Fisher and Arrow-Debreu equilibria. In *ISAAC: International Symposium on Algorithms and Complexity*, pages 647–656, 2009.

[Dev10] N. Devanur. Fisher markets and convex programs. Unpublished manuscript, 2010.

[DPSV08] N. Devanur, C. H. Papadimitriou, A. Saberi, and V. V. Vazirani. Market equilibrium via a primal-dual-type algorithm. *JACM*, 55(5), 2008.

[Eav76] B. C. Eaves. A finite algorithm for the linear exchange model. *Journal of Mathematical Economics*, 3(2):197–203, 1976.

[EG59] E. Eisenberg and D. Gale. Consensus of subjective probabilities: the Pari-Mutuel method. *The Annals of Mathematical Statistics*, 30:165–168, 1959.

[GLS88] M. Grotschel, L. Lovasz, and A. Schirjver. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, 1988.

[GV10] G. Goel and V. V. Vazirani. A perfect price discrimination market model with production, and a (rational) convex program for it. In *3rd International Symposium on Algorithmic Game Theory*, 2010.

[Jai07] K. Jain. A polynomial time algorithm for computing the Arrow-Debreu market equilibrium for linear utilities. *SIACOMP*, 37(1):303–318, 2007.

[JPP00] A. Jalali, R. Padovani, and R. Pankaj. Data throughput of CDMA-HDR a high efficiency-high data rate personal communication wireless system. In *Proceedings of the IEEE Semiannual Vehicular Technology Conference, VTC2000-Spring*, Tokyo, Japan, May 2000.

[JV08] K. Jain and V. V. Vazirani. Eisenberg-Gale markets: Algorithms and game-theoretic properties. *Games and Economic Behavior*, 70(1), 2008.

[Kal77] E. Kalai. Nonsymmetric Nash solutions and replications of 2-person bargaining. *International Journal of Game Theory*, 6(3):129–133, 1977.

[Kal85] E. Kalai. Solutions to the bargaining problem. In L. Hurwicz, D. Schmeidler, and H. Sonnenschein, editors, *Social Goals and Social Organization*, pages 75–105. Cambridge University Press, 1985.

[Kel97] F. P. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8:33–37, 1997.

[Nas50] J. F. Nash. The bargaining problem. *Econometrica*, 18:155–162, 1950.

[NBC$^+$09] N. Nisan, J. Bayer, D. Chandra, T. Franji, R. Gardner, Y. Matias, N. Rhodes, M. Seltzer, D. Tom, and H. Varian. Google's auction for TV ads. In *ICALP*, 2009.

[Nis09] N. Nisan. Report from ALGO 2009. In *Algorithmic Game Theory Blog*, September 10, 2009.

[OR94] M. Osborne and A. Rubinstein. *A Course in Game Theory*. The MIT Press, 1994.

[Orl10] J. B. Orlin. Improved algorithms for computing Fisher's market clearing prices. In *STOC*, 2010.

[TL89] W. Thomson and T. Lensberg. *Axiomatic Theory of Bargaining With a Variable Population*. Cambridge University Press, 1989.

[Tse] D. Tse. Multiuser diversity in wireless networks. http://www.eecs.berkeley.edu/˜dtse/stanford416.ps.

[Vaz07] V. V. Vazirani. Combinatorial algorithms for market equilibria. In N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, editors, *Algorithmic Game Theory*, pages 103–134. Cambridge University Press, 2007.

[Vaz10a] V. V. Vazirani. 2-player Nash and nonsymmetric bargaining games: algorithms and structural properties. In *3rd International Symposium on Algorithmic Game Theory*, 2010.

[Vaz10b] V. V. Vazirani. Spending constraint utilities, with applications to the Adwords market. *Mathematics of Operations Research*, 35(2), 2010.

[Vaz10c] V. V. Vazirani. Seeking combinatorial algorithms for convex programs. In *Noam Nisan's Algorithmic Game Theory Blog*, January 12, 2010.

[VY10] V. V. Vazirani and M. Yannakakis. Market equilibria under separable, piecewise-linear, concave utilities. In *Proceedings of The First Symposium on Innovations in Computer Science*, 2010.

[WGMV95] D.P. Williamson, M.X. Goemans, M. Mihail, and V.V. Vazirani. A primal-dual approximation algorithm for generalized Steiner network problems. *Combinatorica*, 15:435–454, 1995.

A Solution to ADNB in the Limit

Assume that we are given an instance of game **ADNB** that is feasible and let $\mathcal{M}$ be the flexible budget market obtained from it. In this section, we present an algorithm that converges to the equilibrium of $\mathcal{M}$ in the limit.

Algorithm 3 will use the DPSV algorithm as a subroutine [DPSV08]. When this subroutine is called, we assume that the money of each agent is fixed and is specified in the vector $\boldsymbol{m}$.

Let $N'(\boldsymbol{p})$ denote the network for the case that the money of agents is fixed and specified by vector $\boldsymbol{m}$; this network differs from network $N(\boldsymbol{p})$ only in that the capacities of edges going from buyers to t are specified by $\boldsymbol{m}$, rather than being defined as a function of the prices.

Algorithm 3 (Solution to ADNB in the Limit)

```
1. Initialization: $\forall i \in B: \quad m_i \leftarrow 1$.
2. Compute equilibrium prices, $\boldsymbol{p}$, for market $(\boldsymbol{u}, \boldsymbol{m})$ using the DPSV algorithm.
3. For each $i \in B$, compute $\gamma_i$ w.r.t. prices $\boldsymbol{p}$, and set $m'_i \leftarrow 1 + \frac{c_i}{\gamma_i}$.
4. If $\boldsymbol{m}' = \boldsymbol{m}$ then output equilibrium allocations and HALT.
   Else, update $\boldsymbol{m}$ to $\boldsymbol{m}'$ and go to Step 2.
```

Let $\boldsymbol{p}^*$ and $\boldsymbol{m}^*$ be the equilibrium prices and moneys for the flexible budget market $\mathcal{M}$, and let $\boldsymbol{p}^{(k)}$ and $\boldsymbol{m}^{(k)}$ denote the prices and moneys computed by the algorithm in the k-th iteration, $k \geq 1$.

Lemma 37 *$\boldsymbol{p}^{(k)}$ and $\boldsymbol{m}^{(k)}$ are monotone increasing and are weakly dominated by $\boldsymbol{p}^*$ and $\boldsymbol{m}^*$, respectively.*

Proof : We will use the following 2 facts. First, the DPSV algorithm maintains the following invariant throughout:
Invariant: W.r.t. current prices, $\boldsymbol{p}$, $(s, B \cup G \cup t)$ is a min-cut in network $N'(\boldsymbol{p})$.
Second, if $\boldsymbol{p}$ are equilibrium prices for money $\boldsymbol{m}$ and if $\boldsymbol{m'}$ is at least as large as $\boldsymbol{m}$ in each component, then the equilibrium prices for money $\boldsymbol{m'}$ cannot be smaller than $\boldsymbol{p}$ in any component.

Consider the following induction hypothesis:

- the algorithm given above maintains the Invariant throughout.
- $\boldsymbol{p}^{(k)}$ is monotone increasing (hence, for each agent i, γ_i is monotonically decreasing).
- $\boldsymbol{m}^{(k)}$ is monotone increasing.

It is easy to carry out this induction simultaneously for all 3 assertions.

Using the first assertion and Lemma 6, $\boldsymbol{p}^{(k)}$ is weakly dominated bounded by $\boldsymbol{p}^*$. Now, using the formula for money in flexible budget markets, it is easy to see that $\boldsymbol{m}^{(k)}$ is weakly dominated by $\boldsymbol{m}^*$. □

Theorem 38 *Algorithm 3 converges to the equilibrium prices and moneys of market $\mathcal{M}$ in the limit.*

Proof : We will use the following fact: for the linear case of Fisher's model, the analog of Lemma 4 holds, i.e., if $\boldsymbol{p}$ are equilibrium prices for money $\boldsymbol{m}$, then in network $N'(\boldsymbol{p})$, $(s, B \cup G \cup y)$ and $(s \cup B \cup G, t)$ must both be min-cuts (for a proof, see Lemma 5.2 in [Vaz07]).

Since $\boldsymbol{p}^{(k)}$ and $\boldsymbol{m}^{(k)}$ are monotone increasing and bounded, they must converge. Let $\boldsymbol{p}^{(0)}$ and $\boldsymbol{m}^{(0)}$ be their limit points. W.r.t. these prices and moneys, it must be the case that for each $i \in B$, $m_i = 1 + c_i/\gamma_i$ and $(s, B \cup G \cup t)$ and $(s \cup B \cup G, t)$ must both be min-cuts in the corresponding network (by the fact stated above). Using lemma 4 we get that $\boldsymbol{p}^{(0)}$ and $\boldsymbol{m}^{(0)}$ are equilibrium prices and moneys for market $\mathcal{M}$. □

Finally, by Theorem 3 we get:

Corollary 39 *Algorithm 3 converges to the Nash bargaining solution for* **ADNB**.